S01P0067 US00

日本国特許庁
PATENT OFFICE
JAPANESE GOVERNMENT

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２０００年　１月１８日

出願番号
Application Number: 特願２０００－０１３９５７

出願人
Applicant (s): ソニー株式会社

２０００年１１月１７日

特許庁長官
Commissioner,
Patent Office 及川耕造

出証番号　出証特２０００－３０９５２７６

【書類名】　特許願
【整理番号】　9900691003
【提出日】　平成12年 1月18日
【あて先】　特許庁長官殿
【国際特許分類】　H04L 12/40
【発明者】
【住所又は居所】　東京都品川区北品川６丁目７番３５号　ソニー株式会社内
【氏名】　堀口　麻里
【特許出願人】
【識別番号】　000002185
【氏名又は名称】　ソニー株式会社
【代表者】　出井　伸之
【代理人】
【識別番号】　100067736
【弁理士】
【氏名又は名称】　小池　晃
【選任した代理人】
【識別番号】　100086335
【弁理士】
【氏名又は名称】　田村　榮一
【選任した代理人】
【識別番号】　100096677
【弁理士】
【氏名又は名称】　伊賀　誠司
【手数料の表示】
【予納台帳番号】　019530
【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【包括委任状番号】　9707387

【プルーフの要否】　要

【書類名】　　明細書

【発明の名称】　情報処理装置及び方法、媒体

【特許請求の範囲】

【請求項１】　ネットワークを介して他の情報処理装置と接続可能であるとともに、所定の機能を実行する１以上の機能実行手段と、上記機能実行手段の使用予定に関する第１の情報を格納する格納手段とを少なくとも有する情報処理装置において、

上記機能実行手段の使用予定に関する第１の情報に含まれない第２の情報を入手する入手手段と、

上記第２の情報を所定のブロック形式とし、上記第１の情報の付加情報として上記格納手段に格納させる制御手段とを有する

ことを特徴とする情報処理装置。

【請求項２】　上記第１の情報は、所定のディジタルインターフェイスフォーマットにて規定された上記使用予定の情報であり、

上記第２の情報は、上記使用予定の詳細を表す表示可能なテキスト情報であることを特徴とする請求項１記載の情報処理装置。

【請求項３】　上記第２の情報は、ローテキストと当該ローテキストのコーディングを表すキャラクタコード及び言語コードを含むことを特徴とする請求項２記載の情報処理装置。

【請求項４】　上記制御手段は、上記ネットワークに接続された他の情報処理装置の上記格納手段に対して、上記第１の情報及び第２の情報の書き込みと読み出しを制御する機能を有することを特徴とする請求項１記載の情報処理装置。

【請求項５】　上記制御手段は、上記ネットワークに接続された他の情報処理装置の制御手段からの指令に基づいて、上記格納手段に対する上記第１の情報及び上記第２の情報の書き込みと読み出しを制御する機能を有することを特徴とする請求項１記載の情報処理装置。

【請求項６】　上記入手手段は、装置の使用者から入力された上記機能実行手段の使用予定に関する設定情報から上記第２の情報を生成する機能を有することを特徴とする請求項１記載の情報処理装置。

【請求項７】　上記入手手段は、上記ネットワークに接続された他の情報処理装置の格納手段に格納されている上記第２の情報を入手する機能を有することを特徴とする請求項１記載の情報処理装置。

【請求項８】　ネットワークを介して他の情報処理装置と接続可能であるとともに、所定の機能を実行する１以上の機能実行手段と、上記機能実行手段の使用予定に関する第１の情報を格納する格納手段とを少なくとも有する情報処理装置の情報処理方法において、

上記機能実行手段の使用予定に関する第１の情報に含まれない第２の情報を入手し、

上記第２の情報を所定のブロック形式とし、上記第１の情報の付加情報として上記格納手段に格納させる制御を行う

ことを特徴とする情報処理方法。

【請求項９】　上記第１の情報は、所定のディジタルインターフェイスフォーマットにて規定された上記使用予定の情報であり、

上記第２の情報は、上記使用予定の詳細を表す表示可能なテキスト情報であることを特徴とする請求項８記載の情報処理方法。

【請求項１０】　上記第２の情報は、ローテキストと当該ローテキストのコーディングを表すキャラクタコード及び言語コードを含むことを特徴とする請求項９記載の情報処理方法。

【請求項１１】　上記ネットワークに接続された他の情報処理装置の上記格納手段について、上記第１の情報及び第２の情報の書き込みと読み出しを制御することを特徴とする請求項８記載の情報処理方法。

【請求項１２】　上記ネットワークに接続された他の情報処理装置からの指令に基づいて、上記格納手段に対する上記第１の情報及び上記第２の情報の書き込みと読み出しを制御することを特徴とする請求項８記載の情報処理方法。

【請求項１３】　装置の使用者から入力された上記機能実行手段の使用予定に関する設定情報から上記第２の情報を生成することを特徴とする請求項８記載の情報処理方法。

【請求項１４】　上記ネットワークに接続された他の情報処理装置の格納手段に格納されている上記第２の情報を入手することを特徴とする請求項８記載の情報処理方法。

【請求項１５】　１以上の所定の機能の使用予定に関する第１の情報に含まれない第２の情報を入手するステップと、

上記第２の情報を所定のブロック形式とし、上記第１の情報の付加情報として格納手段に格納させるステップとを含むことを特徴とするプログラムを情報処理装置に実行させる媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、例えばＩＥＥＥ１３９４シリアルデータバスを介して他の情報処理装置と接続される情報処理装置において、内蔵するサブユニットを重複することなく確実に制御することができるようにする情報処理装置及び方法、媒体に関する。

【０００２】

【従来の技術】

近年は、例えばＩＥＥＥ（The Institute of Electrical and Electronics Engineers）で規格化されたＩＥＥＥ１３９４シリアルデータバスを用いるネットワークを介して、相互に情報を伝達できるようにしたＡＶ機器が開発されている。このネットワークシステムにおいては、所定のディジタルインターフェイスコマンド（ＡＶ／Ｃ　Command Transaction Set：以下ＡＶ／Ｃコマンドと略称する）を用いて、上述のネットワークに接続されているＡＶ機器を相互に制御することが可能である。

【０００３】

図３４には、ＩＥＥＥ１３９４シリアルデータバスを用いたネットワークの一例を示す。この図３４のネットワークシステムにおいては、ＩＥＥＥ１３９４シリアルデータバス８０（以下、単にバス８０とする）を用いることにより、例えばディジタル衛星放送を受信するＩＲＤ（Integrated Reciever Decoder）７１

で受信された映像信号を、バス８０介して接続されているＤＶＣＲ（Digital Video Cassette Recorder）８１で録画することができる。さらにこれらのＩＲＤ７１、ＤＶＣＲ８１を用いて、いわゆる予約録画をすることも可能である。

【０００４】

この装置で予約録画を行う場合には、例えばＩＲＤ７１内に設けられるコントローラ７２によってＩＲＤ７１自身とＤＶＣＲ８１とが制御される。すなわち、予約録画の設定（チャンネル、開始時刻等の設定）は、ＩＲＤ７１に対して行われる。そして、設定された開始時刻になると、コントローラ７２は、ＩＲＤ７１内のディジタルチューナサブユニット７３に対して内部的に制御し、上記予約されている（設定されている）チャンネルを選局させ、ＣＳアンテナ７４で捉えた信号の中から当該ディジタルチューナサブユニット７３で受信した映像信号等をバス８０を介してＤＶＣＲ８１に出力させる。また同時に、コントローラ７２からは、バス８０を介してＤＶＣＲ８１内に設けられるＶＣＲサブユニット８４に対して録画開始のコマンドが送信される。このときのＶＣＲサブユニット８４は、コントローラ７２から送信されてきた録画開始コマンドに対応して、ディジタルチューナサブユニット７３から送信されてきた映像信号を図示しない磁気テープ等の記録媒体へ記録する。以上により、図３４のネットワークにおける予約録画が行われる。

【０００５】

また、ＤＶＣＲ８１もコントローラ８２を備えており、当該コントローラ８２は、例えば内蔵のアナログチューナサブユニット８３で受信された映像信号をＶＣＲサブユニット８４で記録するなどの制御を行う。

【０００６】

なお、バス８０に接続されているＩＲＤ７１およびＤＶＣＲ８１のような電子機器はユニットと呼ばれており、ユニット間においては、ＡＶ／Ｃコマンド（AV/C Command Transaction Set）の一般的な仕様（AV/C Digital Interface Command Set General Specification、以下、ＡＶ／Ｃジェネラルと略称する）で規定されている記述（Descriptor、以下、適宜ディスクリプタと呼ぶ）を用いて、各ユニットに記憶されている情報を相互に読み書きすることが可能となされている

。ＡＶ／Ｃジェネラルの詳細については、「http://www.1394ta.org/」に公開されている。また、ユニットが有する機能は、サブユニットと呼ばれており、図３４の例では、ＩＲＤ７１のディジタルチューナサブユニット７３、ＤＶＣＲ８１のＶＣＲサブユニット８４等がサブユニットとなる。

【０００７】

【発明が解決しようとする課題】

ところで上述したように、ＤＶＣＲ８１の動作を、バス８０を介して接続されている他の機器（図３４の例の場合、ＩＲＤ７１）が制御できるようなネットワークシステムの場合は、いわゆるダブルブッキングが生ずる虞がある。

【０００８】

例えば、ディジタル衛星放送の録画予約（録画予約Ａとする）をＩＲＤ７１に入力すると、その予約情報は、ＩＲＤ７１のコントローラ７２に記憶されることになる。但し、この時点（録画予約の時間になる前の時点）では、ＩＲＤ７１からＤＶＣＲ８１に対してその録画予約に関連したＡＶ／Ｃコマンドが送信されていないため、当該ＤＶＣＲ８１は、ＩＲＤ７１に録画予約の設定がなされていることを知らない。その後、例えば録画予約Ａの録画時刻に重複する時刻において放送される地上波アナログ放送の録画予約（録画予約Ｂとする）がＤＶＣＲ８１に設定された場合、当該ＤＶＣＲ８１のコントローラ８２は、ＩＲＤ７１に設定されている録画予約Ａに関する情報を得ていないので、時刻が重複しているか否かを知ることができず、上記録画予約Ｂを受け付けて記憶してしまう。

【０００９】

したがって、録画予約Ａおよび録画予約Ｂで指定された時刻になると、ＤＶＣＲ８１のＶＣＲサブユニット８４には、ＩＲＤ７１のディジタルチューナサブユニット７３及びＤＶＣＲ８１のアナログチューナサブユニット８３の両方から映像信号が供給されると共に、ＩＲＤ７１のコントローラ７２及びＤＶＣＲ８１のコントローラ８２の両方による制御がなされてしまうことになり、不都合が生じてしまう。

【００１０】

上述した従来のネットワークシステムにおいて、このようなダブルブッキング

による不都合が発生するのは、例えば、バス８０を介して接続されている各ＡＶ機器（ユニット）が他のＡＶ機器による予約情報等の詳細を入手できないこと、或いは、各ＡＶ機器が他のＡＶ機器に対して自己が持つ予約情報等の詳細を知らせる手段がないこと、また、ユーザが新たな予約の入力を行う場合に既に設定されている予約情報の詳細を知り得ないこと、更には、各ＡＶ機器やユーザが予約情報の詳細を入力したくても入力できないことなどに起因している。

【００１１】

そこで、本発明はこのような状況に鑑みてなされたものであり、ユーザや各機器が予約情報等の詳細を入力できること、各機器やユーザが他の機器による予約情報等の詳細を入手できる（知り得る）こと、及び、各機器が他の機器やユーザに対して自己が持つ予約情報等の詳細を知らせる手段を実現することにより、ダブルブッキングの発生を抑止可能となるだけでなく、分かり易く且つ使い易いネットワークシステムを構築可能とする、情報処理装置及び方法、媒体を提供することを目的とする。

【００１２】

【課題を解決するための手段】

本発明の情報処理装置は、上述した課題を解決するために、ネットワークを介して他の情報処理装置と接続可能であるとともに、所定の機能を実行する１以上の機能実行手段と、上記機能実行手段の使用予定に関する第１の情報を格納する格納手段とを少なくとも有する情報処理装置において、上記機能実行手段の使用予定に関する第１の情報に含まれない第２の情報を入手する入手手段と、上記第２の情報を所定のブロック形式とし、上記第１の情報の付加情報として上記格納手段に格納させる制御手段とを有することを特徴とする。

【００１３】

本発明の情報処理方法は、上述した課題を解決するために、ネットワークを介して他の情報処理装置と接続可能であるとともに、所定の機能を実行する１以上の機能実行手段と、上記機能実行手段の使用予定に関する第１の情報を格納する格納手段とを少なくとも有する情報処理装置の情報処理方法において、上記機能実行手段の使用予定に関する第１の情報に含まれない第２の情報を入手し、上記

第２の情報を所定のブロック形式とし、上記第１の情報の付加情報として上記格納手段に格納させる制御を行うことを特徴とする。

【００１４】

本発明の媒体は、上述した課題を解決するために、１以上の所定の機能の使用予定に関する第１の情報に含まれない第２の情報を入手するステップと、上記第２の情報を所定のブロック形式とし、上記第１の情報の付加情報として格納手段に格納させるステップとを含むことを特徴とするプログラムを情報処理装置に実行させることを特徴とする。

【００１５】

【発明の実施の形態】

本発明の好ましい実施の形態について、図面を参照しながら説明する。なお、本明細書において、システムの用語は、複数の装置、手段などにより構成される全体的な装置を意味するものである。

【００１６】

図１には、本発明実施の形態のネットワークシステムの概略構成を示す。

【００１７】

この図１に示すネットワークシステムは、ＩＥＥＥ１３９４シリアルデータバス２（以下、バス２とする）を介して接続されているＩＲＤ１及びＤＶＣＲ（例えばＤ－ＶＨＳ等のレコーダ）３から構成されている。もちろん、このバス２にはＩＲＤ１やＤＶＣＲ３以外に、例えば、パーソナルコンピュータ、ハードディスクドライブ、ＣＤプレーヤ、モニタ、ディジタルビデオカメラ、ＭＤ（商標）プレーヤ等のＩＥＥＥ１３９４端子を備える電子機器を接続することが可能である。

【００１８】

ＩＲＤ１のコントローラ１１は、ユーザからの選局操作や録画予約操作等を受け付けて、ＩＲＤ１の全体を制御する。また、当該コントローラ１１は、所定のディジタルインターフェイスコマンドとして、前記ＡＶ／Ｃコマンドを用いてＤＶＣＲ３を制御する。ＣＳアンテナ１３は、図示せぬ通信衛星を介して送信されてくるディジタル衛星放送のディジタル信号を受信して、チューナサブユニット

１２に出力する。チューナサブユニット１２は、コントローラ１１の制御に基づいて、ＣＳアンテナ１３から入力されたディジタル信号から所望のチャンネルの信号を抽出し、バス２を介してＤＶＣＲ３のＶＣＲサブユニット３３に出力する。さらに、コントローラ１１は、ＤＶＣＲ３のＢＢＳ（Bulletin Board Subunit）３４に記録されている情報を検索可能となっている。

【００１９】

ＩＲＤ１のサブユニットの一つであるＢＢＳ１４は、コントローラ１１が受け付けて、確定した録画予約等の情報（詳細は図２により後述する）を格納する。

【００２０】

ＤＶＣＲ３のコントローラ３１は、ユーザからの再生指示の操作や録画予約操作等を受け付けて、ＤＶＣＲ３の全体を制御する。また、コントローラ３１は、ＢＢＳ３４に記録されている情報を検索可能である。アナログチューナサブユニット３２は、コントローラ３１の制御に基づいて、入力されるアナログ信号から所定のチャンネルの信号を抽出し、ＶＣＲサブユニット３３に出力する。

【００２１】

ＶＣＲサブユニット３３は、アナログチューナサブユニット３２から入力された映像信号、又は、バス２を介して入力されるＩＲＤ１のチューナサブユニット１２からの映像信号を図示せぬ磁気テープに記録する。

【００２２】

ＢＢＳ３４は、ＤＶＣＲ３に関わる録画予約等の情報（詳細は図２により後述する）を格納する。

【００２３】

この図１に示したネットワークシステムにおいて、ディジタル衛星放送の録画予約を行う場合、ユーザは、ＩＲＤ１に対して録画予約の設定（チャンネル及び録画開始時刻等の設定）を入力する。そして、その録画予約の時刻やチャンネル等が既に予約されているものでない場合、当該入力された録画予約が認められて、その情報がＩＲＤ１のＢＢＳ１４に書き込まれることになる。

【００２４】

ここで、ＩＲＤ１に対してユーザにより録画予約の入力がなされた場合、当該

ＩＲＤ１のコントローラ１１は、図２に示すように、それぞれの録画予約についての内容を表す設定情報としてスケジュールアクション（Scheduled Action１～ｎ）５１を内部のレジスタに保持する。当該スケジュールアクション（Scheduled Action）は、スケジュールデータ（Scheduled Data）５２とプログラム（Program）５３からなる。また、スケジュールデータ（Scheduled Data）５２は、録画の開始時間情報（Start time）、実行時間情報（Duration）、繰り返し情報（Repeat information）、使用するサブユニットの情報（Resources used）からなり、プログラム（Program）５３は、サブユニットを実際に制御するためのコマンド（Command１～ｎ）からなる。上記スケジュールデータ（Scheduled Data）５２の内容（Resource Schedule Object）は、ユーザから入力された録画予約が確定した時に、ＩＲＤ１のコントローラ１１からＤＶＣＲ３へ送られ、当該ＤＶＣＲ３のＢＢＳ３４に書き込まれる。一方、上記プログラム（Program）５３の各コマンドは、実際に録画予約の開始時間になり且つ実行時間になった時に、ＩＲＤ１のコントローラ１１からＤＶＣＲ３のサブユニット（録画を行う際に実際に使用される各サブユニットＡ，Ｂ，・・・，ｎ）に送られることになる。

【００２５】

ＤＶＣＲ３のＢＢＳ３４は、図２に示すように、ＲＳＢ（Resource Schedule Board）６１、及びその他のボード６２から構成される。ＲＳＢ６１は、ＢＢＳ３４内において録画予約の情報のように、リソースの使用予定を書き込むためのリストである。なお、当該ＲＳＢ６１で読み書きされるエントリは使用予定であって、予約された制御を実行するものではない。ＲＳＢ６１には、上記ＩＲＤ１のコントローラ１１より送られてきた上記スケジュールデータ（Scheduled Data）５２のオブジェクト（Start time、Duration、Repeat info、Resources）や、他のユニット（ＤＶＣＲ３のコントローラ３１も含む）より送られてきた各スケジュールデータのオブジェクトが、リソーススケジュールエントリ（Resource Schedule Entry１～ｎ）６３として書き込まれる。

【００２６】

なお、ＤＶＣＲ３のＢＢＳ３４に書き込まれた情報は、ＩＲＤ１のコントローラ１１だけでなく、他のユニットのコントローラ（例えば、ＤＶＣＲ３のコント

ローラ３１も含む）からの要求に対応しても公開される。

【００２７】

図３には、ＢＢＳのＲＳＢのフォーマット構成を示す。

【００２８】

この図３において、descriptor_lengthはＲＳＢの長さを表す。list_typeには、ＲＳＢが読み出し専用（Read Only）であるのか又は書き込み可能（Write Enabled）であるのかが記述される。size_of_list_specific_informationは、list_specific_informationの長さを表し、list_specific_informationは、list_typeによって異なるものとなる。

【００２９】

Write Enabled list_specific_infomationには、図４に示す情報が記述される。Write Enabled list_specific_infomationのnon_info_block_fields_lengthは、non info block fieldsのバイト数を表す。ＲＳＢの場合、board_typeは図５に示すように”01_{16}”とされる。object_list_maximum_sizeは、object listの最大の大きさを表す。object_entry_maximum_numberは、listにおけるobject entriesの最大の数を表す。object_entry_maximum_sizeは、object entryの最大の大きさを表す。以上のobject_list_maximum_size, object_entries_maximum_number, object_entry_maximum_sizeは、それぞれ制限が設けられていない場合には、”0000_{16}”とされる。これらの３つのフィールドは、コントローラがobject listまたは、object entryの容量を知る上において、有意義である。board_type_dependent_information_lengthは、board_type_dependent_informationの長さを表し、board_type_dependent_informationは、board typeに固有の情報を表す。

【００３０】

objec_entryは図６に示すように記述される。objec_entryのdescriptor_lengthは、このdescriptorの長さを表す。entry_typeはBoard entry Descriptorの場合、Boardを表す値”80_{16}”とされる。object_IDは、posting_device_GUIDとrecord_IDとにより構成される。posting_deviceは、ＢＢＳに対して情報を記入（post）したコントローラを意味し、従ってposting_device_GUIDは、そのＧＵＩ

Ｄを表す。record_IDは、ユニット内においてイベント（Event）に対して割当てられたＩＤを表す。size_of_entry_specific_imformationは、entry_specific_informationとしてのResource Schedule Entryの大きさを表す。

【００３１】

Resource Schedule Entryには、図７に示す情報が記述される。

【００３２】

この図７において、non_info_block_lengthは、repeat_informationまでのnon info block fieldsのバイト数を表す。start_timeは、図８に示すように、イベント（event、この例では予約録画）が開始する年月日時分秒を表す。年（year）は１６ビットで表され、西暦を表す４つの数字がそれぞれ４ビットのＢＣＤ（Binary Coded Decimal）で表される。月（month）は８ビットで表され、１（０１）月から１２月までの２桁の数字を表す２つの数字がそれぞれ４ビットのＢＣＤで表される。日（day）は８ビットで表され、１（０１）日から３１日までの２桁の数字（日）を表す２つの数字がそれぞれ４ビットのＢＣＤで表される。時間（hour）は８ビットで表され、０（００時）から２４時までの２桁の数字（時刻）を表す２つの数字がそれぞれ４ビットのＢＣＤで表される。分（minute）は８ビットで表され、０（００）分から６０分までの２桁の数字（分）を表す２つの数字がそれぞれ４ビットのＢＣＤで表される。秒（second）は８ビットで表され、０（００）秒から６０秒までの２桁の数字（秒）を表す２つの数字がそれぞれ４ビットのＢＣＤで表される。このように、start_timeがＢＣＤで表されることにより、その識別が容易となっている。またこの時刻は、ローカルタイムで表される。

【００３３】

イベント（event）の長さを表すDurationは、図９に示すように、時分秒で表される。時間（hours）は３桁の時間を表す３つの数字がそれぞれ４ビットのＢＣＤで表され、合計１２ビットとなる。分（minutes）は２桁の分を表す２つの数字がそれぞれ４ビットのＢＣＤとされ、合計８ビットとなる。秒（seconds）は２桁の秒を表す２つの数字がそれぞれ４ビットのＢＣＤで表され、合計８ビットとされる。start_timeにDurationを加算することで、イベントの終了時刻が表

される。このように、終了時刻を直接的に表現せずに、Durationとしてイベントの長さを表すようにすることで、イベントのstart_timeが変更されたような場合においても、その終了時刻を変更するような操作が不要となり、変更処理が容易となる。

【００３４】

repeat_information_lengthは、repeat_informationの長さを表している。repeat_informationは、いつどのようにしてスケジュール（schedule）が繰り返される（この場合は、予約録画が繰り返される）かを表す。Scheduled Actionが繰り返されない場合、repeat_infomation_lengthは、”００_{16}”とされる。repeat_informationは、選択されたrepeat typeによって内容が異なる。

【００３５】

repeat_typeには、図１０に示すように、Weekly scheduleの”００_{16}”とInterval scheduleの”１０_{16}”とがある。スケジュール（schedule）が、週毎に繰り返される場合、ポスティングデバイス（Posting Device、この例ではＩＲＤ１のコントローラ１１）は、その曜日及び繰り返されるべきイベントの数を図１１に示すように表示する。このrepeat_typeには、図１０に示す値、”００_{16}”が記述される。number_of_eventsには、イベントの数が記述される。日曜日から土曜日までのWeekly flagsは、繰り返されるイベントが開始する週の日を表す。例えば、１３時００分から３時間のイベント（この例では予約録画）が、毎週月曜日と水曜日に開始される場合、月曜日と水曜日のフラグに”１”が設定され、その他のフラグには”０”が設定される。このように、repeat_typeには、週毎に繰り返されるイベントが記録できるので、例えば月曜日と水曜日の絶対的な日時を放送日に対応する分だけ記憶させるような場合に比べて、記憶容量は小さくて済ませることが可能ととなる。

【００３６】

ポスティングデバイス（この例ではコントローラ１１）は、Scheduled Actionが所定の間隔（intervals）で繰り返される場合、図１２に示すようなフォーマットでイベントを記述する。この図１２のrepeat_typeには、この例の場合、図１０に示す値”１０_{16}”が記述される。number_of_eventsには、イベントの数

が記述される。interval（間隔）は、現在のイベントのstart_timeから次のイベントのstart_timeまでの間隔を表す。この間隔は、時分秒で表される。時間は３桁の時間を表す３つの数字がそれぞれ４ビットのＢＣＤで表され、合計１２ビットで表現される。分と秒は、それぞれ２桁の分と秒を表す２つの数字が４ビットのＢＣＤとされ、それぞれ合計８ビットで表現される。このように、repeat_typeには、周期的に繰り返されるイベントが記憶できるので、それぞれが開始される絶対時刻（日時）を別に記憶させる場合に比べて、記憶容量が少なくて済む。

【００３７】

さらに、図７に示すinfo Blocksは、図１３に示すようなフォーマットとされる。

【００３８】

この図１３において、compound_lengthは、このinfo blockのバイト長を表す。ただしlengthフィールド自身は、この長さに含まれていない。info_block_typeは”８９００_{16}”にセットされる。primary_fields_lengthは、number_of_subunitsとsubunit_type_and_ID fieldのバイト数を表す。number_of_subunitsは、ポスティングデバイス（この例ではコントローラ１１）が使用するサブユニットの数を表す。subunit_type_and_IDは、ポスティングデバイスが使用するサブユニットを指定する。

【００３９】

また、本実施の形態の場合、図７に示すResource Schedule Entryには、図１４に示すcharacter_code_information_blockと、図１５に示すlanguage_code_information_blockと、図１６に示すraw_text_information_blockとが記述される。

【００４０】

図１４に示すcharacter_code_information_blockには、図１６のraw_text_information_blockのraw_text_dataが何れのキャラクタコード（character code）でコーディングされているのかを示す情報が記述されている。この図１４において、compound_lengthは、このcharacter_code_information_blockのバイト長を表す。ただし、lengthフィールド自身は、この長さに含まれていない。info_blo

ck_typeはcharacter_code_information_blockであることを示す”０００８$_{16}$”にセットされる。primary_fields_lengthは、character_code_typeとcharacter_code_type_specific fieldのバイト数を表す。character_code_typeはキャラクタコード（character code）の種類を表し、character_code_type_specificはキャラクタコードの仕様を表す。

【００４１】

図１５に示すlanguage_code_information_blockには、図１６のraw_text_information_blockのraw_text_dataが何れの言語コード（language code）でコーディングされているのかを示す情報が記述されている。この図１５において、compound_lengthは、このlanguage_code_information_blockのバイト長を表す。ただし、lengthフィールド自身は、この長さに含まれていない。info_block_typeはlanguage_code_information_blockであることを示す”０００９$_{16}$”にセットされる。primary_fields_lengthは、language_code_typeとlanguage_code_type_specific fieldのバイト数を表す。language_code_typeは言語コード（language code）の種類を表し、language_code_type_specificは言語コードの仕様を表す。

【００４２】

図１６に示すraw_text_information_blockにおいて、compound_lengthは、このraw_text_information_blockのバイト長を表す。ただし、lengthフィールド自身は、この長さに含まれていない。info_block_typeはraw_text_information_blockであることを示す”０００Ａ$_{16}$”にセットされる。primary_fields_lengthは、raw_text_dataのバイト数を表す。raw_text_dataには、図１７に示すように、録画予約に関連する情報として、例えば、チャンネル、プログラム名（番組名）、制御情報（再生、録画、停止等を表すテキスト）、備考（例えばペイパービューの番組等）、プロバイダ名、予約が仮予約である旨の情報などのテキストデータが記述される。

【００４３】

図１に示した本発明実施の形態のネットワークシステムにおいては、各ユニットのＲＳＢのResource Schedule Entryにテキスト情報を書き込むためのinformation blockとして、character_code_information_blockと、language_code_info

rmation_blockと、raw_text_information_blockとを定義しており、これらを用いて、例えば図１７に示したような録画予約に関連する情報など、機器の使用予定に関する様々な付加情報をテキストデータとして書き込み可能とし、また、このテキストデータはネットワークシステムを構成する各ユニットのコントローラにおいて読み取り可能となっている。

【００４４】

したがって、本実施の形態のネットワークシステムにおいては、ユーザや各機器が上記録画予約に関連する情報を入力できること、各機器やユーザが他の機器による上記録画予約に関連する情報を入手できる（知り得る）こと、及び、各機器が他の機器やユーザに対して自己が持つ上記録画予約に関連する情報を知らせることなどが実現可能となっている。

【００４５】

図１８～図２０には、本実施の形態のネットワークシステムにおいて上記録画予約に関連する情報の入力、入手、録画予約に関連する情報の報知を行う際の手順を示す。なお、これら図１８～図２０の処理は、録画予約に関連するリソース、すなわちサブユニットが複数存在する場合、それら各サブユニット毎に順番に行われる処理である。

【００４６】

図１８に示すフローチャートのステップＳ１０からステップＳ１７までは、使用したい機器のＲＳＢが書き込み可能かどうかを調べる手順である。図１のネットワークシステムの例の場合、録画予約に使用するサブユニットは、ＩＲＤ１のチューナサブユニット１２とＤＶＣＲ３のＶＣＲサブユニット３３であるため、ＩＲＤ１のコントローラ１１は、先ず、ＩＲＤ１のＢＢＳ１４内のＲＳＢが書き込み可能であるかどうか調べる。

【００４７】

この図１８において、ユーザから例えば録画日時とチャンネルの録画予約の入力がなされると、ＩＲＤ１のコントローラ１１は、ステップＳ１０の処理として、その予約の情報を例えば内部レジスタに一旦保存する。

【００４８】

コントローラ１１は、上記予約設定入力がなされると、次のステップＳ１１の処理として、上記予約設定の内容に応じて使用するサブユニットが存在する機器内のＲＳＢをライトオープン（WRITE OPEN）させる処理（すなわち書き込み可能な状態にする処理）を実行する。図１の例の場合、予約録画に使用するサブユニットは、ＩＲＤ１のチューナサブユニット１２とＤＶＣＲ３のＶＣＲサブユニット３３であり、ここでは先ず、例えばチューナサブユニット１２を有する機器（ユニット）であるＩＲＤ１のＢＢＳ１４内のＲＳＢをライトオープン（書き込み可能な状態）にする。なお、ＶＣＲサブユニット３３を有する機器であるＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１をライトオープンする処理は、上記ＩＲＤ１のＲＳＢに対して図１８～図２０の処理が行われた後になる。

【００４９】

ここで図２１には、コントローラ１１がＲＳＢをライトオープンする場合に出力するライトオープンコマンド（WRITE OPENコマンド）のフォーマットを示す。但し、ＩＲＤ１のＢＢＳ１４のＲＳＢをライトオープンする場合、ＢＢＳ１４とコントローラ１１とは、同じＩＲＤ１内に設けられているものであり、その間にはバス２は存在していないので、この場合のコントローラ１１は、ＢＢＳ１４に対して、バス２を介してライトオープンコマンドが供給されてきた場合と同様の状態となるようにＢＢＳ１４を制御する。なお、上記ＩＲＤ１のＲＳＢに対して図１８～図２０の処理が行われた後、ＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１をライトオープンする際には、コントローラ１１から図２１に示すフォーマットのライトオープンコマンド（WRITE OPENコマンド）が出力され、バス２を介してＤＶＣＲ３のＢＢＳ３４（実際にはＢＢＳ３４を制御するコントローラ３１）へ送られることになる。

【００５０】

この図２１に示すライトオープンコマンドは、ターゲットの所定のアドレス空間にアクセス（この例ではＢＢＳのＲＳＢにアクセス）するために使用される、オープンディスクリプタコマンド（OPEN DESCRIPTORコマンド）の一種である。この図２１に示すライトオープンコマンドにおいて、opcodeにはオープンディス

クリプタであることを表す値”０８$_{16}$”が記述され、operand０にはライトオープンするディスクリプタの種類を表すdescriptor_typeとして、リストＩＤにより規定されるObject List Descriptorであることを表す値”１０$_{16}$”が記述される。operand１とoperand２には、アクセス先の（ライトオープンする）ＲＳＢのリストＩＤ（この例においては、”００”と”０１”）が記述される。さらにoperand３には、サブファンクション（subfunction）として、ディスクリプタを読み出しまたは書き込みアクセスのためにオープンするライトオープンであることを表す値”０３$_{16}$”が記述される。operand４は、リザーブのための値”００”とされている。

【００５１】

次に、コントローラ１１の処理は、ステップＳ１２に進み、ＩＲＤ１のＢＢＳ１４におけるＲＳＢ内のdescrptior_lengthとlist_specific_information fieldの情報を読み出す。なお、ＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１の読み出し処理は、上記ＩＲＤ１のＲＳＢに対して図１８～図２０の処理が行われた後になる。

【００５２】

図２２には、コントローラ１１がＲＳＢをリード（READ）する場合に出力するリードコマンド（READコマンド）のフォーマットを示す。但し、ＩＲＤ１のＢＢＳ１４のＲＳＢをリードする場合、ＢＢＳ１４とコントローラ１１とは、同じＩＲＤ１内に設けられているものであり、その間にはバス２は存在していないので、この場合のコントローラ１１は、ＢＢＳ１４に対して、バス２を介してリードコマンドが供給されてきた場合と同様の状態となるようにＢＢＳ１４を制御する。なお、上記ＩＲＤ１のＲＳＢに対して図１８～図２０の処理が行われた後、ＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１をリードオープンする際には、コントローラ１１から図２２に示すフォーマットのリードコマンド（READコマンド）が出力され、バス２を介してＤＶＣＲ３のＢＢＳ３４（実際にはＢＢＳ３４を制御するコントローラ３１）へ送られることになる。

【００５３】

図２２に示すリードコマンドのフォーマットにおいて、先頭のopcodeには、リードディスクリプタ（read descriptor）であることを表す値の”０９$_{16}$”が記

述されている。続くoperand０には読み出すディスクリプタ（descriptor）を識別するためのdescriptor identifierが記述される。上記ステップＳ１２における読み出しの処理の場合は、リストＩＤでdescriptor identifierが記述される。具体的には前記図４に示したＲＳＢのwrite enabled list_specific_information fieldのAddress_offsetの”００$_{16}$” 乃至”０Ｄ$_{16}$”が記述される。read_result_statusには、ポスティングデバイスがリードコマンドを送出するときは”ＦＦ$_{16}$”が記述され、ターゲット（この例ではＢＢＳのＲＳＢ）からレスポンスとして返される場合には、読み取り結果が記述される。data_lengthには、ターゲットから読み出されるべきデータのバイト数が記述される。この値が”００$_{16}$”に設定された時、全てのリストが読み出される。addressには、読み出しを開始すべきアドレスが記述される。その値が”００$_{16}$”とされた場合、先頭から読み出しが開始される。

【００５４】

次に、コントローラ１１は、ステップＳ１３の処理として、この時点でのターゲットであるＩＲＤ１のＢＢＳ１４内のＲＳＢに対するリストの最大長の制限（図４におけるobject_list_maximum_size）、リストのエントリ数の制限（図４におけるobject_entries_maximum_number）、および各エントリの最大バイト長の制限（図４におけるobject_entry_maximum_size）を抽出する。

【００５５】

次に、コントローラ１１は、ステップＳ１４の処理として、これからＩＲＤ１のＢＢＳ１４内のＲＳＢにデータ（この場合予約情報）を記録しても、ステップＳ１３で抽出されたリストの最大長（object_list_maximum_size）を超えないか否かを判定する。ステップＳ１４において上記リストの最大長（object_list_maximum_size）を越えないと判定した場合、コントローラ１１の処理はステップＳ１５に進み、一方、越えていると判定した場合、コントローラ１１の処理はステップＳ１７の処理に進む。

【００５６】

ステップＳ１５の処理に進むと、コントローラ１１は、ステップＳ１３で抽出されたリストのエントリ数の制限（最大エントリ数：object_entries_maximum_n

umber）から、現在のエントリ数を引いた値が”０”より大きいか否か、すなわち、まだ記録可能なエントリが残っているか否かを判定する。当該ステップＳ１５において大きい（記録可能なエントリが残っている）と判定された場合、コントローラ１１の処理はステップＳ１６に進み、一方、大きくない（記録可能なエントリが残っていない）と判定された場合、コントローラ１１の処理はステップＳ１７の処理に進む。

【００５７】

ステップＳ１６の処理に進むと、コントローラ１１は、ステップＳ１３で抽出された最大エントリ長（object_entry_maximum_size）から、これから書き込もうとしている予約情報のエントリ長を引いた値が”０”より大きいか否か、すなわち、書き込むべきエントリ長に、まだ余裕があるか否かを判定する。当該ステップＳ１６において大きい（エントリ長に余裕がある）と判定された場合、コントローラ１１の処理は図１９のステップＳ１８に進み、大きくない（エントリ長に余裕がない）と判定された場合、コントローラ１１の処理はステップＳ１７に進む。

【００５８】

ステップＳ１４乃至ステップＳ１６における条件の何れか１つが満足されずにステップＳ１７の処理に進むと、コントローラ１１は、例えば「予約が一杯です」のような警告表示を図示しない表示手段上に行う。すなわち、この場合の警告表示は、ＩＲＤ１のＢＢＳ１４内のＲＳＢに予約情報を書き込むだけの余裕がなくなっていることを表している。これにより、ユーザは、予約が一杯でそれ以上予約をすることができないことを知ることができる。また、ＩＲＤ１のＢＢＳ１４内のＲＳＢの図７に示したResource Schedule Entryに、前述の図１７に示した予約の詳細な内容を表すテキスト情報が既に記述されている場合には、そのテキスト情報の表示を行う。これにより、ユーザは、その予約の詳細な内容を知ることができる。本実施の形態において上述のように予約の詳細な内容を例えばテキスト表示することにより、ユーザは、例えばそれら予約が必要な予約かそうでないか等の判断、すなわち必ず録画したい番組であるか或いは必ずしも録画しなくてもよい番組であるか等の判断ができ、例えば必ず録画したい番組についての

予約はそのまま維持させ、一方、必ずしも録画しなくてもよい番組についてはその予約を取り消し、別の番組の予約を新たに行うなどの、適切な行動を取ることができるようになる。また例えば、予約が仮予約なので上書きしても良いという情報をテキスト情報としてエントリ時に書き込むようにしておけば、後から予約するユーザがより適切な行動、すなわち仮予約されていたものを消して新たな予約を行ったりする等の行動を取ることができるようになる。

【００５９】

一方、ステップＳ１４乃至ステップＳ１６の条件の何れもが満足されている場合、ＩＲＤ１のＢＢＳ１４内のＲＳＢに予約情報を書き込む余裕があるので、コントローラ１１は、図１９のステップＳ１８以降の処理に進み、重複する時刻の予約が既になされているか否かの判定を行う。

【００６０】

すなわち、コントローラ１１は、先ず図１９のステップＳ１８の処理として、変数ｉを”０”に初期設定し、次にステップＳ１９の処理として、ＩＲＤ１のＢＢＳ１４のＲＳＢに記録されているエントリの数（number_of_entries）から変数ｉを減算した値が”０”より大きいか否か、すなわちＢＢＳ１４のＲＳＢに記録されている全てのエントリについて検索が行われたか否かを判定する。このステップＳ１９の処理において、number_of_entriesから変数ｉを減算した値が”０”より大きいと判定した場合、まだ検索していないエントリが存在するので、コントローラ１１の処理はステップＳ２０に進み、ＢＢＳ１４のＲＳＢに掲示されている前記図６のobject_entry［ｉ］を読み出す（この場合はobject_entry［０］を読み出す）。なお、当該ステップＳ２０における読み出しも、前記図２２に示したリードコマンドで行われるが、この場合におけるdescriptor identifierの記述はobject positionで行われる。このobject entry［ｉ］には、既に登録されている予約の時刻情報（図７におけるstart_time, Duration）や、その予約において、使用するサブユニットの識別情報（図１３におけるsubunit_type_and_ID［０］）、前記図１４～図１７に示したテキスト情報などが記憶されている。

【００６１】

次に、コントローラ１１は、ステップＳ２１の処理として、ステップＳ１０でユーザより入力された時刻情報（start_time, Duration）が、ステップＳ２０で読み出された時刻情報（start_time, Duration）と重複しているか否かを判定する。このステップＳ２１において、時刻が重複していると判定された場合、コントローラ１１はステップＳ２２の処理に進み、一方、重複していないと判定した場合、コントローラ１１はステップＳ２３の処理に進む。

【００６２】

ステップＳ２１にて時刻が重複していると判定してステップＳ２２の処理に進むと、コントローラ１１は、ステップＳ１０で予約設定されたサブユニット（この場合、チューナサブユニット１２）が、ステップＳ２０で読み出したサブユニット（subunit_type_and_ID）と一致しているか否かを判定する。当該ステップＳ２２にてサブユニットが一致していると判定した場合、コントローラ１１の処理はステップＳ２４に進み、一方、サブユニットが一致していないと判定した場合、コントローラ１１の処理はステップＳ２３に進む。

【００６３】

ステップＳ２２にてサブユニットが一致していると判定された場合は、結局、時刻とサブユニットの両方が一致していることになるので、コントローラ１１はステップＳ２４の処理として、例えば「予約が重なっています」のような警告表示を行う。これにより、ユーザは、予約が重なることを知ることができて予約のダブルブッキングの発生を防止できることになる。また、ＩＲＤ１のＢＢＳ１４のＲＳＢの図７に示したResource Schedule Entryに、テキスト情報として予約の設定内容が記述されている場合は、そのテキスト情報の表示を行う。これにより、ユーザは、その重なる予約の詳細な内容を知ることができる。本実施の形態において予約の詳細な内容を例えばテキスト表示することにより、ユーザは、例えばそれら予約が必要な予約かそうでないか等の判断ができ、例えば必要な予約についてはそのまま維持させ、また必ずしも必要でない予約を取り消し、別の新たな予約を行うなどの適切な行動を取ることができるようになる。また例えば、予約が仮予約なので上書きしても良いという情報をテキスト情報としてエントリ

時に書き込むようにしておけば、後から予約するユーザがより適切な行動、すなわち仮予約されていたものを消して新たな予約を行ったりする等の行動を取ることができるようになる。

【００６４】

一方で、ステップＳ２１において、時刻が重複していないと判定された場合、又は、時刻が一致していたとしてもステップＳ２２においてサブユニットが一致していないと判定された場合、予約のダブルブッキングが発生する恐れはないので、コントローラ１１は、ステップＳ２３の処理として、変数ｉを１だけインクリメントした後、ステップＳ１９に戻り、number_of_entriesから変数ｉを減算した値が０より大きくないと判定されるまで、同様の処理を繰り返し実行する。すなわち、コントローラ１１では、ＩＲＤ１のＢＢＳ１４のＲＳＢに記憶されている全てのobject entry［ｉ］について、重複する時刻の予約がなされているか否かの検索を行う。

【００６５】

また、ステップＳ１９において、number_of_entriesから変数ｉを減算した値が”０”より大きくないと判定された場合（全てのobject entry［ｉ］の検索が終了した場合）、コントローラ１１の処理は、図２０のステップＳ２５に進む。なお、前述したステップＳ１４乃至ステップＳ１７の処理は、ステップＳ１９の処理において、ＮＯと判定された場合に実行するようにすることも可能である。

【００６６】

図２０のステップＳ２５の処理に進むと、コントローラ１１は、ＩＲＤ１のＢＢＳ１４のＲＳＢのobject entryをクリエイト（create）する。なお、ＤＶＣＲ３のＢＢＳ３４のＲＳＢのクリエイト処理は、上記ＩＲＤ１のＲＳＢに対して図１８～図２０の処理が行われた後になる。

【００６７】

図２３には、コントローラ１１がＲＳＢのobject entryをクリエイト（create）する場合に出力するクリエイトコマンド（CREATEコマンド）のフォーマットを示す。但し、ＩＲＤ１のＢＢＳ１４のＲＳＢをクリエイトする場合、ＢＢＳ１４とコントローラ１１とは、同じＩＲＤ１内に設けられているものであり、その間

にはバス２は存在していないので、この場合のコントローラ１１は、ＢＢＳ１４に対して、バス２を介してクリエイトコマンドが供給されてきた場合と同様の状態となるようにＢＢＳ１４を制御する。なお、上記ＩＲＤ１のＲＳＢに対して図１８～図２０の処理が行われた後、ＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１のobject entryをクリエイトする際には、コントローラ１１から図２３に示すフォーマットのクリエイトコマンド（CREATEコマンド）が出力され、バス２を介してＤＶＣＲ３のＢＢＳ３４（実際にはＢＢＳ３４を制御するコントローラ３１）へ送られることになる。

【００６８】

また、図２４には、図２３内のsubfunction_1で指定できる値を示し、本実施の形態では”０１”（create a new object and its child list）が使用される。また、図２５は、図２３内のsubfunction_1_specification for subfunction_1＝０１のフォーマットを示す。さらに図２６は、図２５内の各フィールド値を示した図である。図２５のdescriptor_identifier_where, descriptor_identifie_what_1,_2の各フイールドに、図２６に示すように、”20_{16}”，”22_{16}”，”11_{16}”をそれぞれ設定すると、”create a newobject and its child list”の意味となる。

【００６９】

これらＡＶ／Ｃコマンドにおけるクリエイトコマンドについての詳細は、ＩＥＥＥ１３９４（インターネットホームページhttp://www.1394TA.org参照）に記述されているものであり、本実施の形態中の各図は、その文献（Enhancement to the AV/C General Specification 3.0 Version 1.0 FC2や、TA Document 1999005 AV/C Bulletion Board Subunit General Specification 1.0 Draft 0.99:149）中のものを記載してある。また、ボードを構成するインフォメーションディスクリプタ（Information List Descriptor）にも書き込み可能なものと読み出し可能なものがあり、これらの区別には、リストタイプが使用される。

【００７０】

なお、外部からＡＶ／Ｃディスクリプタ（AV/C Descriptor）に新規に情報を書き込む方法の１つとして、例えばコントローラ（ポスティングデバイス）がタ

ーゲットに対して前述したクリエイトコマンドを発行し、当該ターゲットが情報を書き込む雛形を作った後、再度、コントローラが具体的な内容を書き込む制御を行うような方法が一般的な方法として考えられる。例えば、初めて情報を書き込む場合、コントローラは所望のリストを指定して、ＡＶ／Ｃディスクリプタクリエイトコマンド（AV/C Descriptor CREATE command）を発行する。このコマンドを受けたターゲットでは、ＡＶ／Ｃジェネラルで指定されたデータ構造の雛形に基づいたオブジェクトをターゲットの内部に作ることになる。また、ＡＶ／Ｃジェネラルで決められたデータ構造の雛形には、オブジェクトＩＤを示すフィールドがある。ＡＶ／Ｃディスクリプタ（AV/C Descriptor）を用いたリストでは、オブジェクトＩＤはターゲットが管理することになる。つまり、オブジェクトをクリエイト（CREATE）した段階で、ターゲットがそのオブジェクトを一意に指定できるＩＤを付け、そのＩＤを管理する機能をターゲットが所有することになる。

【００７１】

オブジェクトＩＤとは、リスト内でそのオブジェクトを一意に指定するための識別番号であり、このため当該オブジェクトＩＤを重複しないようにする機能が管理する側に必要になる。ＢＢＳ自体は情報を提供する場所であり、オブジェクトＩＤの管理はコントローラが持つことになる。

【００７２】

ところが、サブユニットに対してクリエイトコマンドが発行された時、矛盾が生ずる虞がある。すなわち、オブジェクトをクリエイト（CREATE）した際には、コントローラが管理すべきオブジェクトＩＤを、ターゲットが割り振ることになるからである。また、クリエイトコマンドの発行後は、ライト制御を続けて行う必要がある。このように、処理が複数ステップにわかれていることにより、コントローラが書き込み途中に例えばバスから外されたような場合には、不完全なオブジェクトが作成されてしまう可能性がある。

【００７３】

したがって、上記のような状況においては、その不完全なオブジェクトを特定し、そのようなオブジェクトができたときに、当該オブジェクトを良好に削除で

きるシステムが必要になる。

【００７４】

そこで、本発明の実施の形態では、ＢＢＳへの書き込み手段を規格で規定し、不完全なオブジェクトを特定できる仕組みを用意している。

【００７５】

すなわち、先ず、ターゲットは、オブジェクトＩＤ（グローバルユニークＩＤ（Grobal Unique ID：GUID）とrecord IDとで構成される）のうちのＧＵＩＤの部分に、一時的に管理する番号（例えば、全て”０”）を割り振るようにする。コントローラは、先にオブジェクト内部に情報を書き込み、正常に書き込みが終了したならば、最後にＧＵＩＤを書き換える。

【００７６】

上記の手順を決めることにより、正常に書き込み作業が終了したときには、ＧＵＩＤが全て”０”となっているオブジェクトはできないことになり、したがって、ＧＵＩＤが全て”０”のオブジェクトは、書き込み途中で不完全となったオブジェクトと特定できることになる。

【００７７】

これにより、書き込み途中のオブジェクトを一意に特定することができ、また、正常に書き込まれたオブジェクトと不完全なオブジェクトとを区別でき、さらに不完全なオブジェクト（無効なオブジェクト）を簡単に削除することが可能となる。このことにより、電子機器に設けられている有限なメモリを有効活用できるようになる。また、書き込み途中のオブジェクトの特定方法は、オブジェクトＩＤのＧＵＩＤ部分を全て”０”にするような簡単な方法なので、不完全なオブジェクトを削除するためのソフトウェアの作成も容易となる。

【００７８】

なお、図２０のステップＳ２５において、上記クリエイトコマンドを利用する代わりに、インサートコマンド（INSERTコマンド）を使用することも可能である。

【００７９】

次に、ステップＳ２６の処理に進むと、コントローラ１１は、ＩＲＤ１のＢＢ

Ｓ１４のＲＳＢのentry_specific_information fieldsの部分（図２、図７）に予約内容を書き込む。すなわち、これにより、start_time, Duration, repeat_information, 使用するサブユニット（subunit_type_and_ID）などが書き込まれる。

【００８０】

図２７は、このような場合にコントローラ１１が出力するライトディスクリプタコマンド（WRITE DESCRIPTORコマンド）のフォーマットを表している。但し、ＩＲＤ１のＢＢＳ１４のＲＳＢに書き込みを行う場合、ＢＢＳ１４とコントローラ１１とは同じＩＲＤ１内に設けられているため、この場合のコントローラ１１は、ＢＢＳ１４に対して、バス２を介してライトディスクリプタコマンドが供給されてきた場合と同様の状態となるようにＢＢＳ１４を制御する。なお、上記ＩＲＤ１のＲＳＢに対して図１８～図２０の処理が行われた後、ＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１に書き込みを行う際には、コントローラ１１から図２７に示すフォーマットのライトディスクリプタコマンドが出力され、バス２を介してＤＶＣＲ３のＢＢＳ３４（実際にはＢＢＳ３４を制御するコントローラ３１）へ送られることになる。

【００８１】

図２７において、先頭のopcodeには、WRITE DESCRIPTORであることを表す値”$０Ａ_{16}$”が記述される。operand０には、書き込み対象となるディスクリプタを識別させるためのdescriptor indentifierが記述される。この記述は、object positionで行われる。以下、subfunctionとして、partial_replaceであることを表す値”５０_{16}”が記述される。これによりパーシャルインサート（partial insert）またはパーシャルディレート（partial delate）が実行される。インサート（insert）では、descriptor_identifierで指定されるoperandにより規定される１つ前に新しいディスクリプタ（descriptior）が挿入される。デリート（delete）ではdescriptior_identifierにより規定されるディスクリプタが削除される。group_tagは、ライトディスクリプタコマンドが発行される必要があるディスクリプタ上において、分割できない更新の処理を行うために利用される。この例においては、データをディスクリプタに迅速に書き込むことを表す値”００_{16}

”（immediate）が記述されている。replacement_data_lengthは、replacement_dataのoperandにおけるバイト数、すなわち書き込みたいデータの長さを表している。addressは、処理が行われるべき位置を表している。replacement_data_lengthの”０”はパーシャルデリート（partial delete）を意味し、その場合、replacement_dataのoperandは存在しない。この場合、originalにdata_lengthは”０”より大きい値となり、それが削除されるべきバイトの数を表す。original_data_lengthが”０”である場合、パーシャルインサート（partial insert）処理が行われる。この場合、replacement_data_lengthは、”０”より大きい値とされ、挿入されるべきバイトの数を表している。

【００８２】

次に、コントローラ１１は、ステップＳ２７の処理として、ＩＲＤ１のＢＢＳ１４内のＲＳＢの図７に示したResource Schedule Entryに、テキスト情報として予約の設定内容を追加（記述）するか否か判定し、追加すると判定した場合はステップＳ２８の処理に進み、追加しないと判定した場合はステップＳ３１の処理に進む。

【００８３】

ステップＳ２８に進むと、コントローラ１１は、先ず図２８に示すSubunit Identifier Descriptor内のＲＳＢボードタイプを読み出す。

【００８４】

図２８において、descriptor_Lengthは、このディスクリプタ（Descriptor）の長さを表す。generation_IDは、このＢＢＳにおいて、どのＡＶ／Ｃディスクリプタ（AV/C descriptor）フォーマットが使用されるかを表し、通常”００_{16}”とされる。size_of_list_IDは、list IDのバイト数を表す。size_of_object_IDは、このobject IDのバイト数を表す。size_of_object_posionは、list内のオブジェクト（object）の位置が参照されるときに使用されるバイト数を表す。number_of_root_object_listsは、このＢＢＳが直接関連するroot object listsの数を表す。root_object_list_id_x（x＝０，１，２，・・・，ｎ－１）は、このＢＢＳが関連するroot object listsのそれぞれのＩＤを表す。subunit_dependent_information_lengthはsubunit_dependent_informationの長さを表し、subuni

t_dependent_informationには、このＢＢＳが依存するフォーマット及びコンテンツに関する情報が記述される。subunit_dependent_informationには、non_info_blocにfiedlds_length, bulletin_board_subunit_version, number_of_supported_board_type (n) , supported_board_type_specific_of_length [０] の他、supported_board_type_specific_info [０] 乃至supported_board_type_specific_info [n-1] と、それらの長さを表すsupported_board_type_specific_of_length [０] 乃至supported_board_type_specific_of_length [n-1] が含まれている。さらに、ＢＢＳには、manufacturer_dependent_informationの長さを表す、manufacturer_dependent_lengthと製造者に依存する情報を含むmanufacturer_dependent_informationが記述される。root_object_list_idの値は、それがＲＳＢを表すものである場合、図２９に示すように、所定の値”１００１_{16}”とされる。このように、ＲＳＢ（図２９ではResource Schedule Listと記述されている）を表すＩＤを、所定の値に固定しておくことにより、ＲＳＢを読み出す処理が容易となる。

【００８５】

また、図２８のsupported_board_type_specific_informationフィールドは、図３０に示すようなフォーマットとされる。この図３０において、supported_board_typeには、図５に示したＲＳＢであることを表す値”０１_{16}”が記述される。supported_board_type_versionは、bulletin Bond Type Specificationのバージョンの番号を表す。inplementation_profile_IDはこのboard typeのためのprofile IDバージョンを表す。supported_board_type_dependend_information_lengthは、supported_type_dependent_informationのバイト数を表す。supported_board_type_dependent_informationには、各board type specificationに固有の情報が記述される。

【００８６】

次に、ステップＳ２９の処理に進み、コントローラ１１は、ＢＢＳ１４のＲＳＢのバージョンが１．０、すなわちＲＳＢのResource Schedule Entryにテキスト情報を書き込むためのinformation blockとして、前記図７及び図１４～図１７に示したcharacter_code_information_blockと、language_code_information_

blockと、raw_text_information_blockとが定義されているバージョン１．０よりも、ＲＳＢのバーションが大きいか否か判定し、大きいと判定したときはステップＳ３０の処理に進み、大きくないと判定したときはステップＳ３１の処理に進む。

【００８７】

ステップＳ３０の処理に進むと、コントローラ１１は、ユーザからの予約設定情報の入力に応じたテキスト情報を、上記ＢＢＳ１４のＲＳＢのResource Schedule Entry内のテキスト情報を書き込むためのinformation blockに追加する。

【００８８】

次に、ステップＳ３１の処理に進むと、コントローラ１１は、リスト、すなわちＢＢＳ１４のＲＳＢをクローズする。図３１には、ＲＳＢをクローズする場合にコントローラ１１が出力するクローズコマンド（CLOSEコマンド）のフォーマットを表している。但し、ＩＲＤ１のＢＢＳ１４のＲＳＢをクローズする場合、ＢＢＳ１４とコントローラ１１とは同じＩＲＤ１内に設けられているため、この場合のコントローラ１１は、ＢＢＳ１４に対して、バス２を介してクローズコマンドが供給されてきた場合と同様の状態となるようにＢＢＳ１４を制御する。なお、上記ＩＲＤ１のＲＳＢに対して図１８～図２０の処理が行われた後、ＤＶＣＲ３のＢＢＳ３４のＲＳＢのクローズを行う際には、コントローラ１１から図３１に示すフォーマットのクローズコマンドが出力され、バス２を介してＤＶＣＲ３のＢＢＳ３４（実際にはＢＢＳ３４を制御するコントローラ３１）へ送られることになる。

【００８９】

この図３１に示すクローズコマンドのフォーマットは、基本的に図２１に示したライトオープンコマンドと同様のフォーマットであり、subfunctionが、図２１においては、ライトオープン（WRITE OPEN）を表す”03_{16}”とされているのに対して、図３１のクローズコマンドにおいては、クローズ（CLOSE）であることを表す値”00_{16}”とされている点が異なっている。その他の構成は図２１における場合と同様である。

【００９０】

次にステップＳ３２に進み、コントローラ１１は、予約に関連する他のリソースが存在するか否かを判定する。この場合、これまでの処理によりＩＲＤ１のチューナサブユニット１２についての予約の処理は終了しているが、ＤＶＣＲ３のＶＣＲサブユニット３３についての予約の処理は行われていないため、当該ステップＳ３２において他のリソースがあると判定され、図１８のステップＳ１１の処理に戻る。

【００９１】

これ以降、コントローラ１１は、当該他のリソースとしてＤＶＣＲ３のＶＣＲサブユニット３３についての予約の処理を実行する。すなわち、コントローラ１１は、当該ＤＶＣＲ３のＳＳＢ３４のＲＳＢ６１に対して、前述同様に、ステップＳ１１以降の処理を行うことになる。

【００９２】

当該ＤＶＣＲ３のＶＣＲサブユニット３３についての予約処理を実行する場合のコントローラ１１は、ステップＳ１１の処理として、ＶＣＲサブユニット３３を有する機器であるＤＶＣＲ３のＢＢＳ３４内のＲＳＢ６１を、ライトオープン（書き込み可能な状態）するために、前記図２１に示したライトオープンコマンドを、バス２を介してＤＶＣＲ３のＢＢＳ３４（実際にはＢＢＳ３４を制御するコントローラ３１）へ送る。

【００９３】

次に、コントローラ１１は、ステップＳ１２の処理として、前記図２２に示したリードコマンドをバス２を介してＤＶＣＲ３のＢＢＳ３４（実際にはＢＢＳ３４を制御するコントローラ３１）へ送り、ＲＳＢ６１のdescrptior_lengthとlist_specific_information fieldの情報を読み出す。

【００９４】

次に、コントローラ１１は、ステップＳ１３の処理として、ＤＶＣＲ３のＢＢＳ３４内のＲＳＢ６１に対するリストの最大長の制限（図４におけるobject_list_maximum_size）、リストのエントリ数の制限（図４におけるobject_entries_maximum_number）、および各エントリの最大バイト長の制限（図４におけるobjec

t_entry_maximum_size）を抽出する。

【００９５】

次に、コントローラ１１は、前記ステップＳ１４～ステップＳ１６の処理を行う。ステップＳ１４乃至ステップＳ１６における条件の何れか１つが満足されずにステップＳ１７の処理に進むと、コントローラ１１は、例えば「予約が一杯です」のような警告表示を図示しない表示手段上に行う。すなわち、この場合の警告表示は、ＤＶＣＲ３のＢＢＳ３４内のＲＳＢ６１に予約情報を書き込むだけの余裕がなくなっていることを表している。これにより、ユーザは、予約が一杯でそれ以上予約をすることができないことを知ることができる。また、ＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１の図７に示したResource Schedule Entryに、前述の図１７に示した予約の詳細な内容を表すテキスト情報が既に記述されている場合には、そのテキスト情報の表示を行う。これにより、ユーザは、その予約の詳細な内容を知ることができる。本実施の形態において上述のように予約の詳細な内容を例えばテキスト表示することにより、ユーザは、例えばそれら予約が必要な予約かそうでないか等の判断、すなわち必ず録画したい番組であるか或いは必ずしも録画しなくてもよい番組であるか等の判断ができ、例えば必ず録画したい番組についての予約はそのまま維持させ、一方、必ずしも録画しなくてもよい番組についてはその予約を取り消し、別の番組の予約を新たに行うなどの、適切な行動を取ることができるようになる。また例えば、予約が仮予約なので上書きしても良いという情報をテキスト情報としてエントリ時に書き込むようにしておけば、後から予約するユーザがより適切な行動、すなわち仮予約されていたものを消して新たな予約を行ったりする等の行動を取ることができるようになる。

【００９６】

一方、ステップＳ１４乃至ステップＳ１６の条件の何れもが満足されている場合、ＤＶＣＲ３のＢＢＳ３４内のＲＳＢ６１に予約情報を書き込む余裕があるので、コントローラ１１は、図１９のステップＳ１８以降の処理に進み、重複する時刻の予約が既になされているか否かの判定を行う。すなわち、コントローラ１１は、図１９のステップＳ１８の処理として変数ｉを初期設定し、次にステップＳ１９の処理として、ＢＢＳ３４のＲＳＢ６１に記録されているエントリの数（

number_of_entries）から変数ｉを減算した値が”０”より大きいか否かを判定する。このステップＳ１９の処理において、number_of_entriesから変数ｉを減算した値が”０”より大きいと判定した場合、まだ検索していないエントリが存在するので、コントローラ１１はステップＳ２０の処理として、ＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１に掲示されている前記図６のobject_entry［ｉ］を読み出す。

【００９７】

次に、コントローラ１１は、ステップＳ２１の処理として、ステップＳ１０でユーザより入力された時刻情報（start_time, Duration）が、ステップＳ２０で読み出された時刻情報（start_time, Duration）と重複しているか否かを判定する。

【００９８】

ステップＳ２１にて時刻が重複していると判定してステップＳ２２の処理に進むと、コントローラ１１は、ステップＳ１０で予約設定されたサブユニット（この場合、ＶＣＲサブブユニット３３）が、ステップＳ２０で読み出したサブユニット（subunit_type_and_ID）と一致しているか否かを判定する。

【００９９】

ステップＳ２２にてサブユニットが一致していると判定された場合は、結局、時刻とサブユニットの両方が一致していることになるので、コントローラ１１はステップＳ２４の処理として、例えば「予約が重なっています」のような警告表示を行う。これにより、ユーザは、予約が重なることを知ることができて予約のダブルブッキングの発生を防止できることになる。また、ＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１の図７に示したResource Schedule Entryに、テキスト情報として予約の設定内容が記述されている場合は、そのテキスト情報の表示を行う。これにより、ユーザは、その重なる予約の詳細な内容を知ることができる。本実施の形態において予約の詳細な内容を例えばテキスト表示することにより、ユーザは、例えばそれら予約が必要な予約かそうでないか等の判断ができ、例えば必要な予約についてはそのまま維持させ、また必ずしも必要でない予約を取り消し、別の新たな予約を行うなどの適切な行動を取ることができるようになる。また例

えば、予約が仮予約なので上書きしても良いという情報をテキスト情報としてエントリ時に書き込むようにしておけば、後から予約するユーザがより適切な行動、すなわち仮予約されていたものを消して新たな予約を行ったりする等の行動を取ることができるようになる。

【０１００】

一方で、ステップＳ２１において、時刻が重複していないと判定された場合、又は、時刻が一致していたとしてもステップＳ２２においてサブユニットが一致していないと判定された場合、予約のダブルブッキングが発生する恐れはないので、コントローラ１１は、ステップＳ２３の処理として、変数ｉを１インクリメントした後、ステップＳ１９に戻り、number_of_entriesから変数ｉを減算した値が０より大きくないと判定されるまで、同様の処理を繰り返し実行する。すなわち、コントローラ１１では、ＤＶＣＲ３ＢＢＳ３４のＲＳＢ６１に記憶されている全てのobject entry［ｉ］について、重複する時刻の予約がなされているか否かの検索を行う。

【０１０１】

また、ステップＳ１９において、number_of_entriesから変数ｉを減算した値が”０”より大きくないと判定された場合（全てのobject entry［ｉ］の検索が終了した場合）、コントローラ１１は、図２０のステップＳ２５の処理として、前記図２３に示したクリエイトコマンドをバス２を介してＤＶＣＲ３のＢＢＳ３４（実際にはＢＢＳ３４を制御するコントローラ３１）へ送り、当該ＢＢＳ３４のＲＳＢ６１のobject entryをクリエイトする。

【０１０２】

次に、コントローラ１１は、ステップＳ２６の処理として、前記図２７に示したライトディスクリプタコマンドをバス２を介してＤＶＣＲ３のＢＢＳ３４（実際にはＢＢＳ３４を制御するコントローラ３１）へ送り、当該ＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１のentry_specific_information fieldsの部分（図２、図７）に予約内容を書き込む。これにより、start_time, Duration, repeat_information, 使用するサブユニット（subunit_type_and_ID）などが書き込まれる。

【０１０３】

次に、コントローラ１１は、ステップＳ２７の判定処理において、ＤＶＣＲ３のＢＢＳ３４内のＲＳＢ６１の図７に示したResource Schedule Entryに、テキスト情報として予約の設定内容を追加すると判定した場合、ステップＳ２８の処理として、図２８に示したSubunit Identifier Descriptor内のＲＳＢボードタイプを読み出し、次のステップＳ２９にて、ＢＢＳ３４のＲＳＢ６１のバージョンが１．０、すなわちＲＳＢのResource Schedule Entryにテキスト情報を書き込むためのinformation blockとして、図７及び図１４～図１７に示したcharacter_code_information_blockと、language_code_information_blockと、raw_text_information_blockとが定義されているバージョン１．０よりも、ＲＳＢのバージョンが大きいか否か判定する。

【０１０４】

ステップＳ２９にてバージョン１．０より大きいと判定してステップＳ３０に進むと、コントローラ１１は、ユーザからの予約設定情報の入力に応じたテキスト情報を、上記ＲＳＢ６１のResource Schedule Entryのinformation blockに追加する。

【０１０５】

次に、ステップＳ３１の処理に進むと、コントローラ１１は、図３１に示したクローズコマンドをバス２を介してＤＶＣＲ３のＢＢＳ３４（実際にはＢＢＳ３４を制御するコントローラ３１）へ送り、当該ＢＢＳ３４のＲＳＢ６１をクローズする。

【０１０６】

その後ステップＳ３２に進み、コントローラ１１は、予約に関連する他のリソースが存在するか否かを判定する。この場合、これまでの処理によりＩＲＤ１のチューナサブユニット１２とＤＶＣＲ３のＶＣＲサブユニット３３についての予約の処理は終了しており、他に予約処理を行うべきサブユニットがないため、処理を終了する。

【０１０７】

次に、オブジェクトＩＤ設定処理について、図３２のフローチャートを参照し

て説明する。この処理は、ユーザがＩＲＤ１に対して、ＤＶＣＲ３を用いる録画予約の操作を入力し、その操作がコントローラ１１に検知され、上述した図１８乃至図２０の予約処理が実行されて、その録画予約が認められた後、開始される。この例においては、所定の時刻から所定の時間の間、実行されるイベント（予約処理）に関し、それを識別するために、オブジェクトＩＤが対応される。このオブジェクトＩＤは、７２ビットで構成され、そのうちのＭＳＢ側の６４ビットは、その予約に関する全ての情報を持っている機器の固有のＩＤ、具体的にはその機器のＧＵＩＤ（Global Unique ID)とされ、ＬＳＢ側の８ビットは、その機器内で設定される固有の値とされ、具体的にはレコードＩＤ（record ID）とされる。このようにオブジェクトＩＤをＧＵＩＤとレコードＩＤとで構成することにより、イベントを識別するためのオブジェクトＩＤを設定すると、ＩＤ設定の処理が容易となる。

【０１０８】

すなわちこのオブジェクトＩＤは、バス２に接続されている各機器の間において、識別可能である必要がある。そうでなければ、この例においては、１つのユニット（機器）のＲＳＢの内容を他のユニット（機器）が読み込み可能であり、複数のユニットのサブユニットにおいて、共同して（関連して）処理が実行されることがあるので、関連する処理があるか否かを、他のユニットが識別できる必要があるからである。

【０１０９】

バス２に接続されている、ユニット間において、識別可能なものであれば、例えば、オブジェクトＩＤをイベントが発生する毎に、番号１から順番に順次設定するようにすることも可能である。しかしながらそのようにすると、所定のユニットが、所定のイベントに関して、オブジェクトＩＤを設定しようとした場合、バス２に接続されている全てのユニットが、既に設定しているオブジェクトＩＤを読み出して、競合するものがあるか否かを判定し、競合しないものに設定する必要が生じる。しかしながらこの例においては、ＧＵＩＤがオブジェクトＩＤに含まれており、ＧＵＩＤの値は、ユニット毎に異なるものであることが保証されている。したがって各ユニットは、レコードＩＤを自分自身の内部において、競

合するものがないように設定すれば、ＧＵＩＤとレコードＩＤを組み合わせて得られるオブジェクトＩＤは、他のユニットが設定したオブジェクトＩＤと競合することはない。そこで各ユニットは、図３２のフローチャートに示すような処理を行って、ＲＳＢに登録するイベントに対するオブジェクトＩＤを設定する。

【０１１０】

ステップＳ４１において、コントローラ１１は、オブジェクトＩＤ（＝Global Unique ID+record ID）のうちのレコードＩＤとして、ＩＲＤ１内において、その予約情報を一意に識別させるための仮ＩＤを発生する。

【０１１１】

ステップＳ４２において、コントローラ１１は、ＩＲＤ１のＢＢＳ１４内のＲＳＢに登録されているイベントの中から、１つのイベントを抽出して、そのイベントに対応するオブジェクトＩＤの中のレコードＩＤを抽出する。ステップＳ４３において、コントローラ１１は、ステップＳ４１で発生した仮ＩＤと、ステップＳ４２で抽出したレコードＩＤが一致するか否かを判定し、一致する場合、ステップＳ４１に戻り、仮ＩＤを変更し、一致しない仮ＩＤが発生されるまで、ステップＳ４１乃至Ｓ４３の処理を繰り返す。

【０１１２】

仮ＩＤとステップＳ４２で抽出したレコードＩＤが一致しないと判定された場合、ステップＳ４４において、コントローラ１１は、関連するサブユニットに関するＲＳＢに公開されている全てのイベントを抽出したか否かを判定し、まだ抽出していないイベントが残っている場合はステップＳ４２に戻り、全てのイベントを読み出したと判定するまで、ステップＳ４２乃至Ｓ４４の処理を繰り返す。全てのイベントを読み出したと判定された場合、ステップＳ４５において、コントローラ１１は、ＩＲＤ１の機器ＩＤ（ＧＵＩＤ）に、ステップＳ４１で発生した仮ＩＤ（レコードＩＤ）を付加して、オブジェクトＩＤを生成し、ＲＳＢに記述する。

【０１１３】

このように、オブジェクトＩＤは、関連するサブユニットに関するＲＳＢをチェックするだけで（関連しないサブユニットを有するユニットのＲＳＢをチェッ

クすることなく）設定することができる。これは、上述したように、他のユニットのＲＳＢにおけるオブジェクトＩＤは、そのユニットのＧＵＩＤが含まれているため、他のユニットが設定したオブジェクトＩＤは、自分自身が設定したオブジェクトＩＤと一致することが有り得ないからである。したがって、オブジェクトＩＤを簡単に設定することが可能となる。

【０１１４】

本実施の形態において、上述した一連の処理は、ハードウェアにより実行させることもできるが、ソフトウェアにより実行させることもできる。一連の処理をソフトウェアにより実行させる場合には、そのソフトウェアを構成するプログラムが、専用のハードウェアとしてのコントローラに組み込まれているか若しくは当該プログラムがインストールされた、例えば汎用のパーソナルコンピュータなどにより各種の機能を実行することが可能となる。

【０１１５】

汎用のパーソナルコンピュータ１０１は、例えば、図３３に示すように、ＣＰＵ（Central Processing Unit）１１１を内蔵している。ＣＰＵ１１１には、バス１１５を介して入出力インターフェース１１６が接続されており、ＣＰＵ１１１は、入出力インターフェース１１６を介して、ユーザから、キーボード、マウスなどよりなる入力部１１８から指令が入力されると、それに対応して、ＲＯＭ（Read Only Memory）１１２あるいはハードディスク１１４などの記録媒体、または、ドライブ１２０に装着された磁気ディスク１３１，光ディスク１３２、光磁気ディスク１３３などの記録媒体から、それらに記録されている、上述した一連の処理を実行するプログラムを読み出し、ＲＡＭ（Ramdom Access Memory）１１３に書き込み、実行する。なお、ハードディスク１１４に格納されているプログラムには、予め格納されてユーザに配布されるものだけでなく、衛星もしくは、ネットワークから転送され、通信郁１１９により受信されてダウンロードされたプログラムも含まれる。

【０１１６】

また、ＣＰＵ１１１は、プログラムの処理結果のうち、画像信号を、入出力インターフェース１１６を介して、ＬＣＤ（LiquID Crystal Display），ＣＲＴ

（Cathode Ray Tube）などよりなる表示部１１７に出力する。

【０１１７】

なお、本発明は、上述の説明した実施の形態に限定されるものではなく、本発明の精神を逸脱することなく種々の変形が可能とされるものである。例えば本実施の形態では、ＩＲＤ１のコントローラ１１が、ＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１への前記テキスト情報の書き込みと読み出しを制御しているが、さらに他の機器（ユニット）のコントローラが当該ＤＶＣＲ３のＢＢＳ３４のＲＳＢ６１へのテキスト情報の書き込みと読み出しを制御することも可能である。このとき、本実施の形態によれば、例えばＤＣＶＲ等で確認画面を出すような場合に、予約の内容をＲＳＢのinformation blockの形式としているので、他の機器による予約の概要を、上記ＤＶＣＲの確認画面上で、本体装置や外部装置の区別無く、同じ情報量でユーザに表示することができる。

【０１１８】

【発明の効果】

本発明の情報処理装置及び方法、媒体においては、機能実行手段の使用予定に関する第１の情報に含まれない第２の情報を入手し、第２の情報を所定のブロック形式とし、第１の情報の付加情報として格納手段に格納させる制御を行うことにより、ユーザや各機器が予約情報等の詳細を入力でき、また、各機器が他の機器による予約情報等の詳細を入手でき、及び、各機器が他の機器に対して自己が持つ予約情報等の詳細を知らせることを実現可能となり、その結果、ダブルブッキングの発生を抑止可能となるだけでなく、分かり易く且つ使い易いネットワークシステムを構築可能である。

【０１１９】

すなわち、本発明によれば、例えば予約のダブルブッキングが起きるときの警告表示を出すときに、そのダブルブッキングの発生原因をより分かり易い形でユーザに知らせることができる。また、本発明によれば、例えば仮の予約を可能とし、仮予約なので上書きしても良いという情報をエントリ時に書き込むことにより、後から予約するユーザがより適切な行動、つまる仮予約されていたものを消して新たな予約を行ったりする等の行動を取ることができるようになる。さらに

、例えばＤＣＶＲ等で確認画面を出す場合は、その予約の概要を表示することが一般に行われるが、本発明によれば、予約の内容（第２の情報）を所定のブロック形式とすることで、他の機器による予約の概要を、確認画面上で本体装置、外部装置の区別無く、同じ情報量でユーザに表示することができる。

【図面の簡単な説明】

【図１】

本発明実施の形態のネットワークシステムの構成例を示すブロック図である。

【図２】

ターゲットデバイスのＢＢＳのＲＳＢをポスティングデバイスが制御する際の説明に用いる図である。

【図３】

ＢＢＳのＲＳＢのフォーマットを説明する図である。

【図４】

図３のWrite Enabled list_specific_informationのフォーマットを説明する図である。

【図５】

図４のboard_typeのフォーマットを説明する図である。

【図６】

図３のobject_entryのフォーマットを説明する図である。

【図７】

図６のResource Schedule Entryのフォーマットを説明する図である。

【図８】

図７のstart_timeのフォーマットを説明する図である。

【図９】

図７のDurationのフォーマットを説明する図である。

【図１０】

図７のrepeat_typeのフォーマットを説明する図である。

【図１１】

スケジュールが週毎に繰り返される場合のrepeat_informationのフォーマット

を説明する図である。

【図１２】

スケジュールが所定の間隔で行われる場合のrepeat_informationのフォーマットを説明する図である。

【図１３】

図７のInfo blocksのフォーマットを説明する図である。

【図１４】

図７に示すResource Schedule Entryのcharacter_code_information_blockのフォーマットを説明する図である。

【図１５】

図７に示すResource Schedule Entryのlanguage_code_information_blockのフォーマットを説明する図である。

【図１６】

図７に示すResource Schedule Entryのraw_text_information_blockのフォーマットを説明する図である。

【図１７】

図１６に示すraw_text_information_block内に記述されるraw_text_dataの一例を説明する図である。

【図１８】

図１のネットワークシステムの動作のうち、使用したい機器のＲＳＢが書き込み可能かどうか調べる手順を説明するフローチャートである。

【図１９】

図１のネットワークシステムの動作のうち、使用予定のリソースが他の機器から使われる予定になっているかどうか確認する手順を説明するフローチャートである。

【図２０】

図１のネットワークシステムの動作のうち、使用予定を書き込む時の手順を説明するフローチャートである。

【図２１】

ライトオープンコマンドのフォーマットを説明する図である。

【図２２】

リードコマンドのフォーマットを説明する図である。

【図２３】

クリエイトコマンドのフォーマットを説明する図である。

【図２４】

図１２のsubfunction_1を説明する図である。

【図２５】

図２４のsubfunction_1の詳細を説明する図である。

【図２６】

図２５におけるフィールドの値の例を示す図である。

【図２７】

ライトディスクリプタコマンドのフォーマットを説明する図である。

【図２８】

ＢＢＳのフォーマットを説明する図である。

【図２９】

ＢＢＳのroot_object_list_IDを説明する図である。

【図３０】

図２８のＢＢＳのsupported_board_type_specific_informationのフォーマットを説明する図である。

【図３１】

クローズコマンドのフォーマットを説明する図である。

【図３２】

オブジェクトＩＤ設定処理を説明するフローチャートである。

【図３３】

コンピュータの構成例を示すブロック図である。

【図３４】

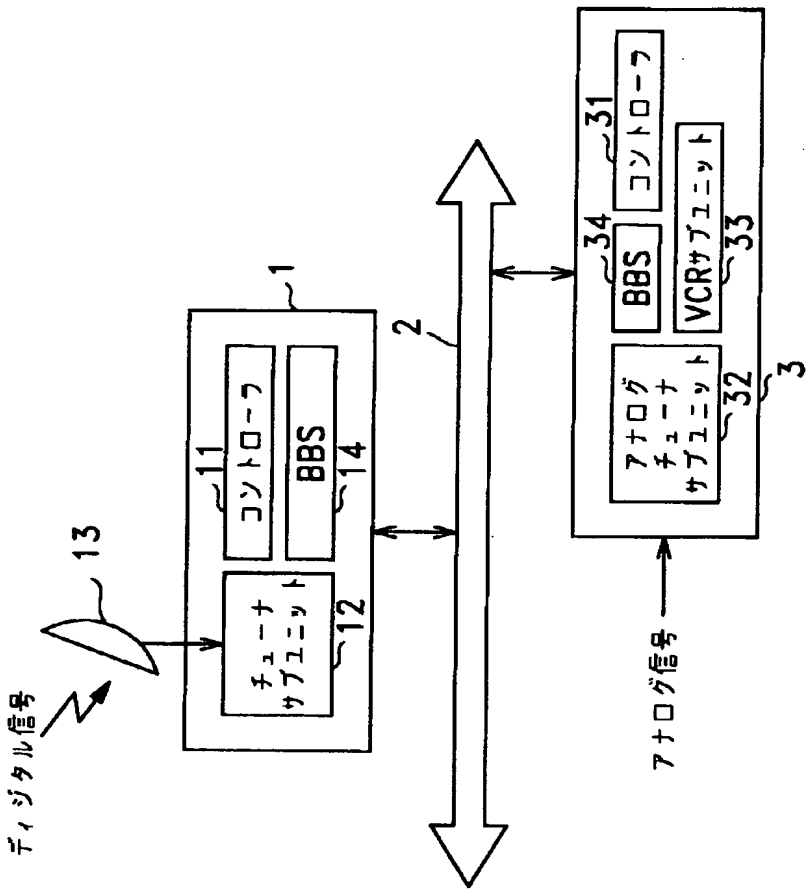
従来のネットワークシステムの構成例を示すブロック図である。

【符号の説明】

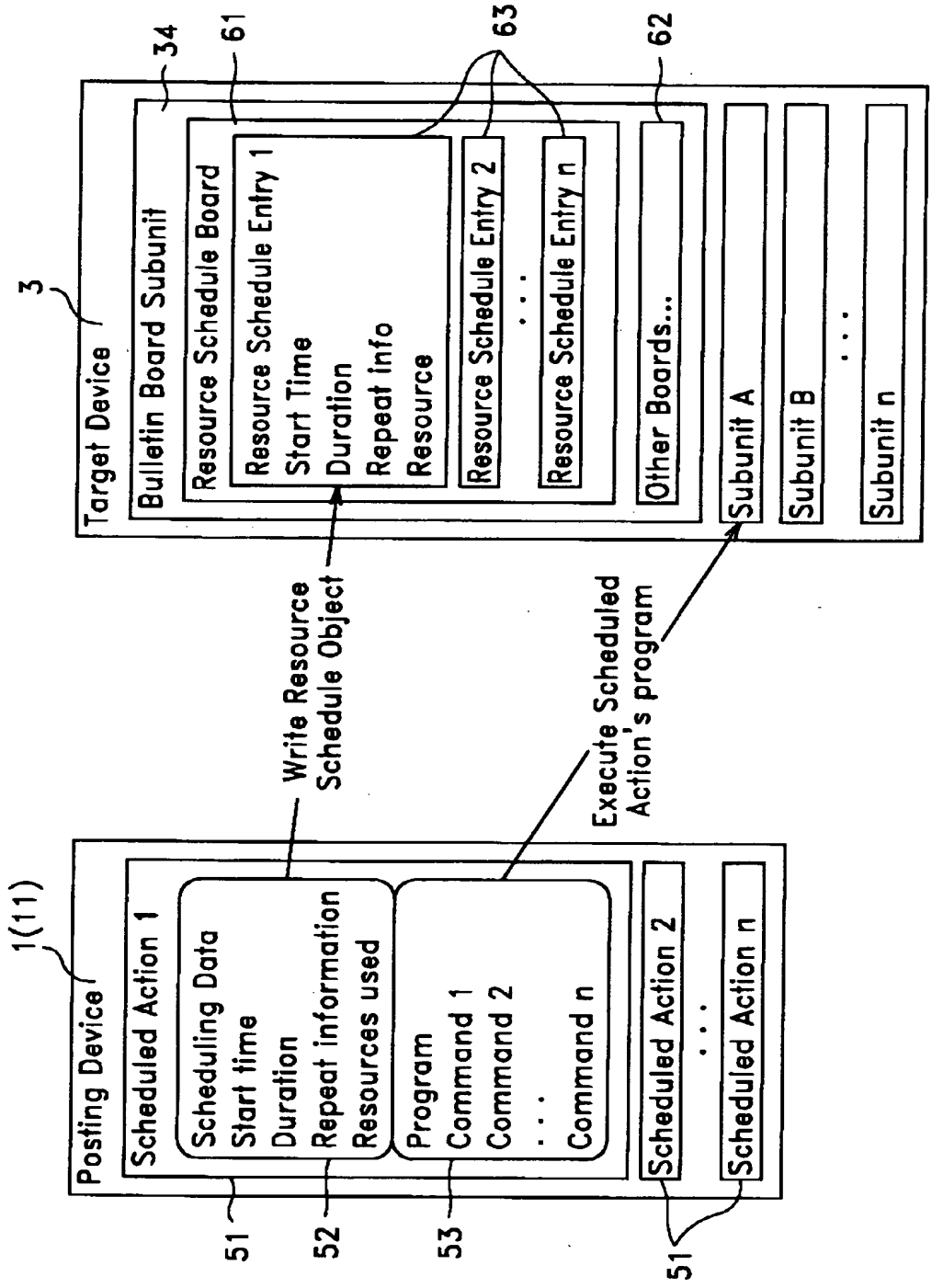
１　ＩＲＤ、　２　ＩＥＥＥ１３９４シリアルデータバス、　３　ＤＶＣＲ、　１１　コントローラ、　１２　チューナサブユニット、　１４　ＢＢＳ、　３１　コントローラ、　３２　アナログチューナサブユニット、　３３　ＶＣＲサブユニット、　３４　ＢＢＳ、　６１　ＲＳＢ

【書類名】　　図面

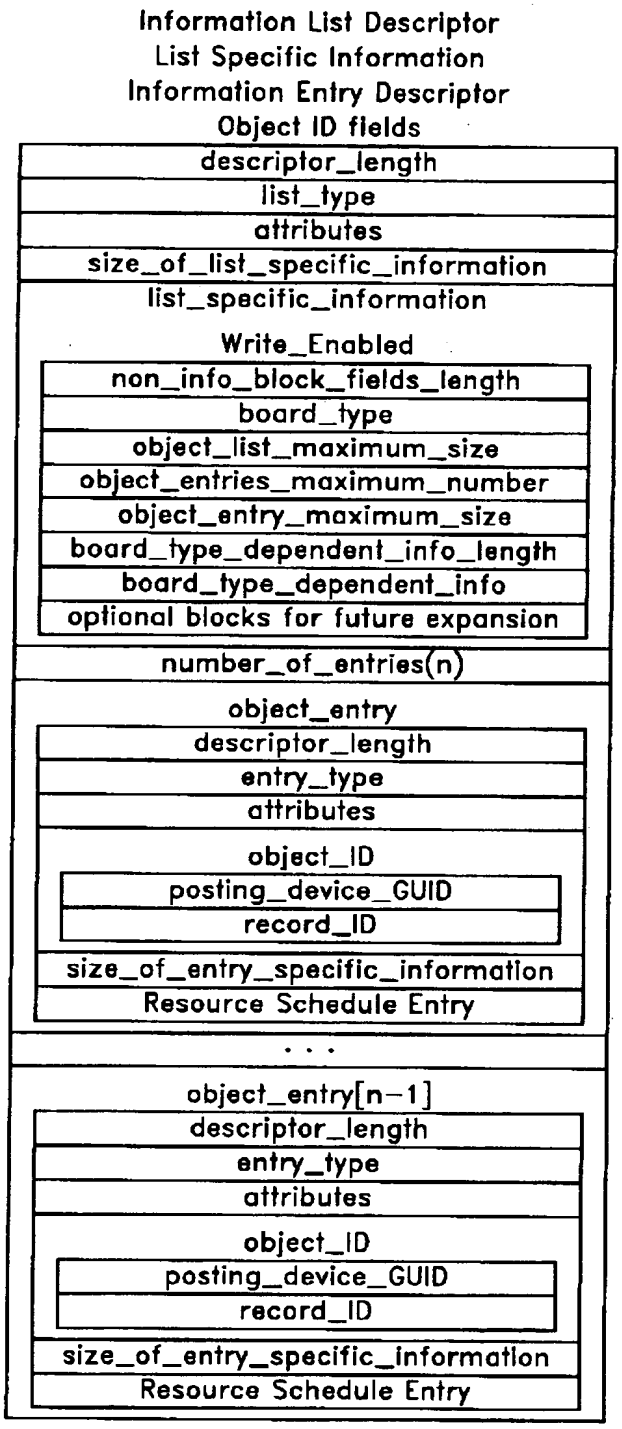
【図１】


1
2
3
11
コントローラ
BBS
14
チューナ
サブユニット
12
13
ディジタル信号
31
コントローラ
34
BBS
VCRサブユニット
33
アナログ
チューナ
サブユニット
32
アナログ信号

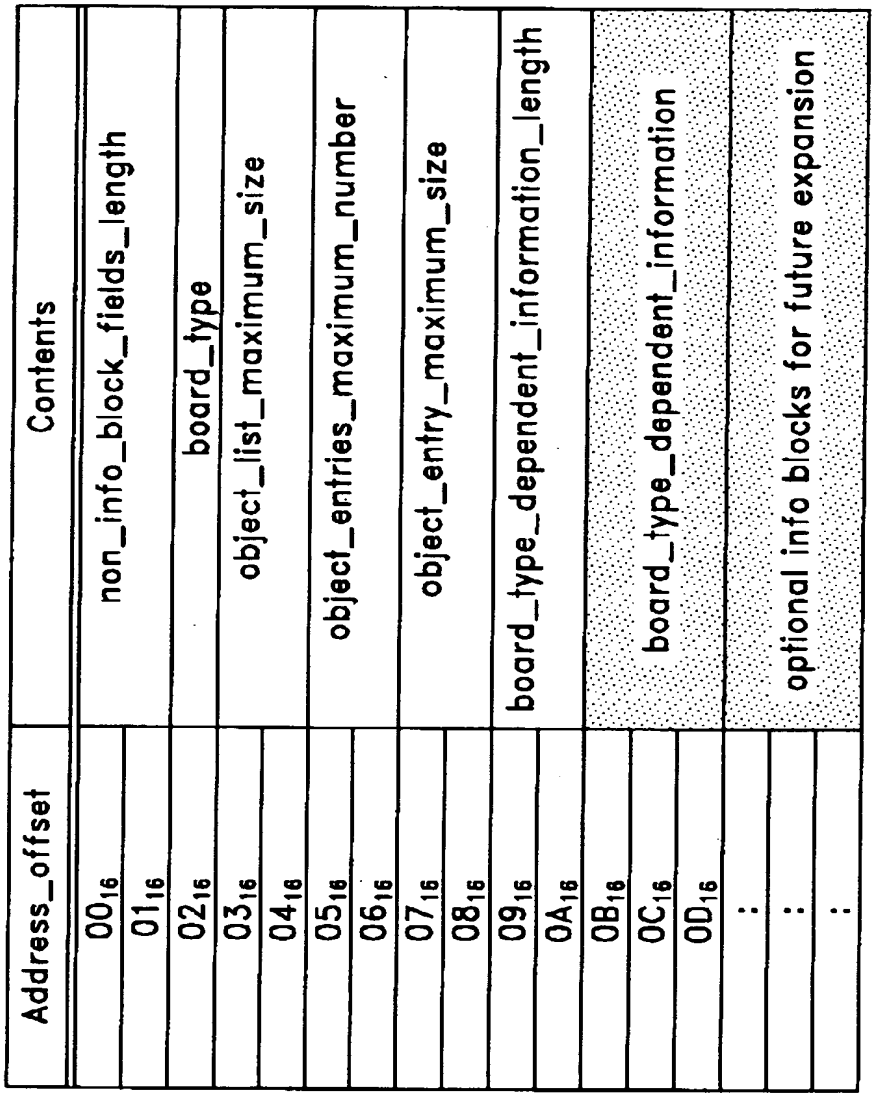
【図２】


Target Device
3
Bulletin Board Subunit
34
Resource Schedule Board
61
Resource Schedule Entry 1
Start Time
Duration
Repeat Info
Resource
Resource Schedule Entry 2
Resource Schedule Entry n
63
Other Boards...
62
Subunit A
Subunit B
Subunit n
Write Resource Schedule Object
Execute Scheduled Action's program
Posting Device
1(11)
Scheduled Action 1
Scheduling Data
Start time
Duration
Repeat information
Resources used
Program
Command 1
Command 2
Command n
Scheduled Action 2
Scheduled Action n
51
52
53
51

【図３】


Information List Descriptor
List Specific Information
Information Entry Descriptor
Object ID fields
descriptor_length
list_type
attributes
size_of_list_specific_information
list_specific_information
Write_Enabled
non_info_block_fields_length
board_type
object_list_maximum_size
object_entries_maximum_number
object_entry_maximum_size
board_type_dependent_info_length
board_type_dependent_info
optional blocks for future expansion
number_of_entries(n)
object_entry
descriptor_length
entry_type
attributes
object_ID
posting_device_GUID
record_ID
size_of_entry_specific_information
Resource Schedule Entry
. . .
object_entry[n-1]
descriptor_length
entry_type
attributes
object_ID
posting_device_GUID
record_ID
size_of_entry_specific_information
Resource Schedule Entry

【図４】



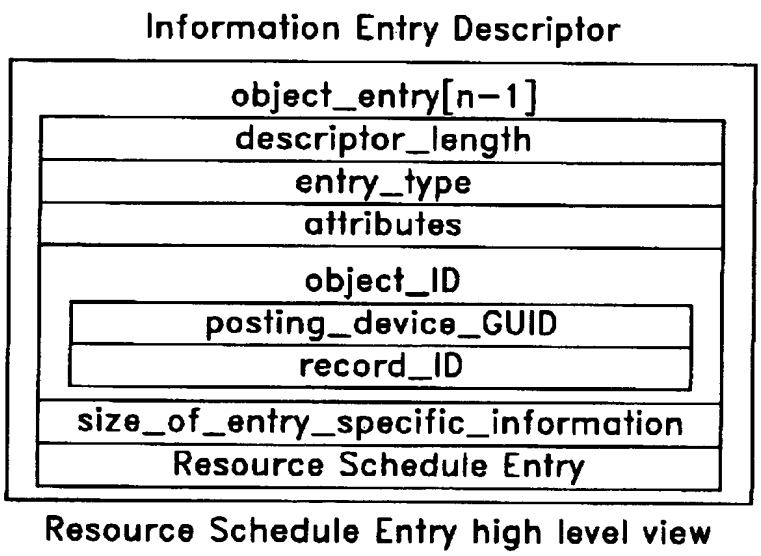
Write enabled list_specific_information fields for the Information List Descriptor

【図５】

Existing board types

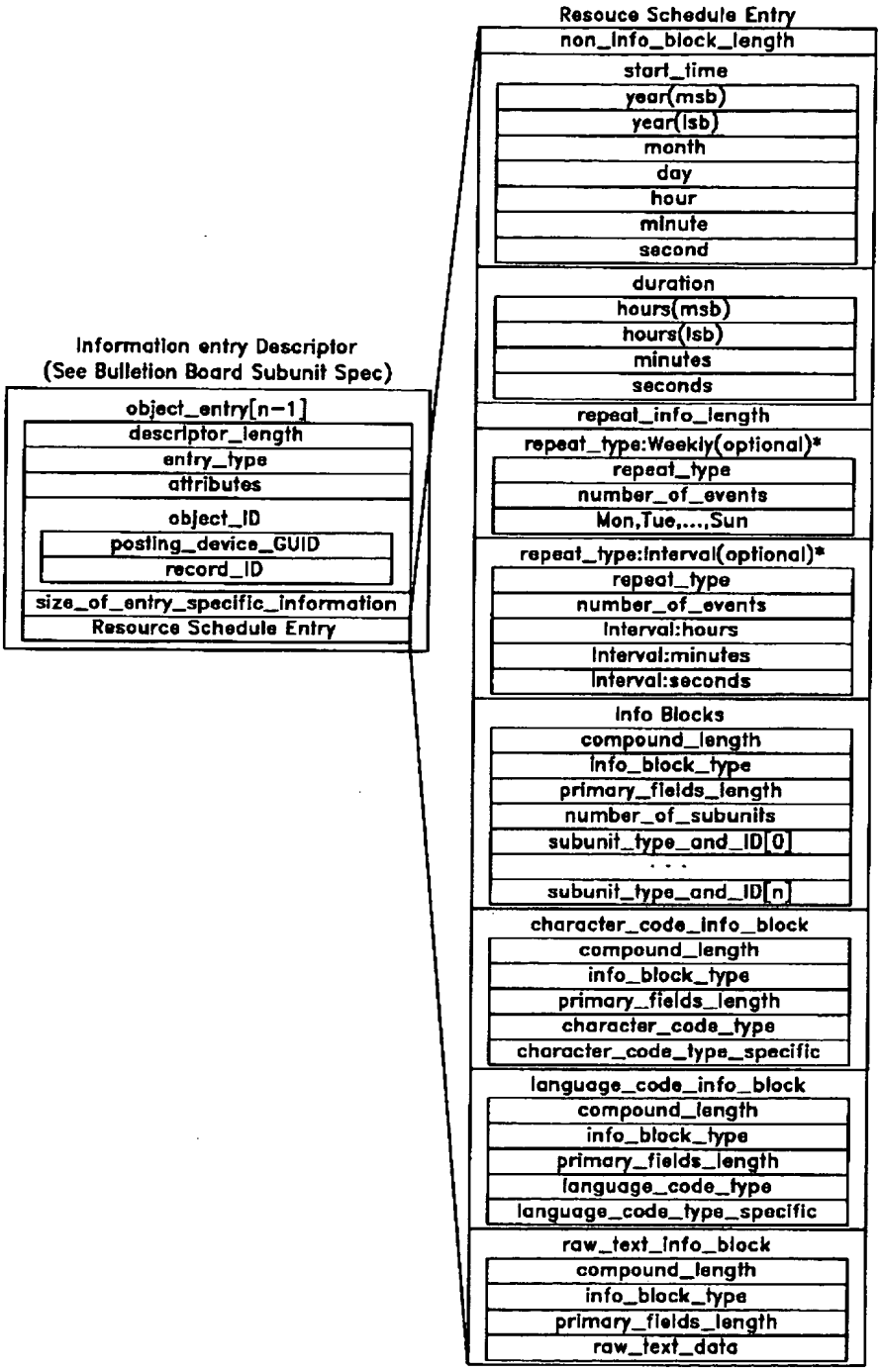
Value	Board type
00_{16}	Reserved
01_{16}	Resource Schedule Board
02_{16}–FF_{16}	Reserved for future specification

【図６】



Resource Schedule Entry high level view

【図７】


Information entry Descriptor
(See Bulletion Board Subunit Spec)
object_entry[n-1]
descriptor_length
entry_type
attributes
object_ID
posting_device_GUID
record_ID
size_of_entry_specific_information
Resource Schedule Entry
Resouce Schedule Entry
non_info_block_length
start_time
year(msb)
year(lsb)
month
day
hour
minute
second
duration
hours(msb)
hours(lsb)
minutes
seconds
repeat_info_length
repeat_type:Weekly(optional)*
repeat_type
number_of_events
Mon,Tue,...,Sun
repeat_type:Interval(optional)*
repeat_type
number_of_events
Interval:hours
Interval:minutes
Interval:seconds
Info Blocks
compound_length
info_block_type
primary_fields_length
number_of_subunits
subunit_type_and_ID[0]
subunit_type_and_ID[n]
character_code_info_block
compound_length
info_block_type
primary_fields_length
character_code_type
character_code_type_specific
language_code_info_block
compound_length
info_block_type
primary_fields_length
language_code_type
language_code_type_specific
raw_text_info_block
compound_length
info_block_type
primary_fields_length
raw_text_data

【図８】

Address_offset	Contents
00_{16}	year(msb)
01_{16}	year(lsb)
02_{16}	month
03_{16}	day
04_{16}	hour
05_{16}	minute
06_{16}	second

start_time fields for Resouce Schedule Entries

【図９】

Address_offset	Contents	
00_{16}	Reserved(4 bits)	hours(msb)
01_{16}	hours(lsb)	
02_{16}	minutes	
03_{16}	seconds	

duration fields for Resouce Schedule Entries

【図１０】

repeat_type value assignment

Values	definition
00_{16}	Weekly schedule
01_{16}–$0F_{16}$	reserved
10_{16}	Interval schedule
$0F_{16}$–FF_{16}	reserved

【図１１】

	msb							lsb
address_offset	contents							
$0E_{16}$	repeat_type							
$0F_{16}$	number_of_events							
10_{16}	Sunday	Monday	Tuesday	Wed-nesday	Thurs-day	Friday	Saturday	Re-served

repeat_information fields for Weekly

【図１２】

address_offset	contents	
$0E_{16}$	repeat_type	
$0F_{16}$	number_of_events	
10_{16}	Reserved(4 bits)	Interval:hours(msb)
11_{16}	interval:hours(lsb)	
12_{16}	interval:minutes	
13_{16}	interval:seconds	

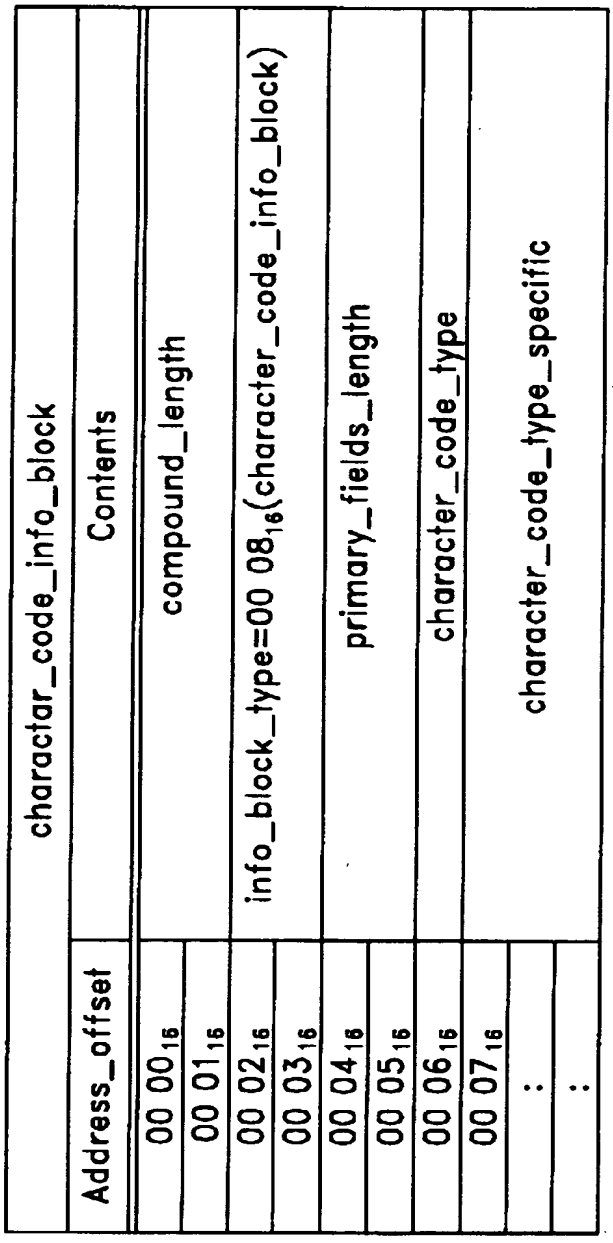
repeat_information fields for interval

【図１３】

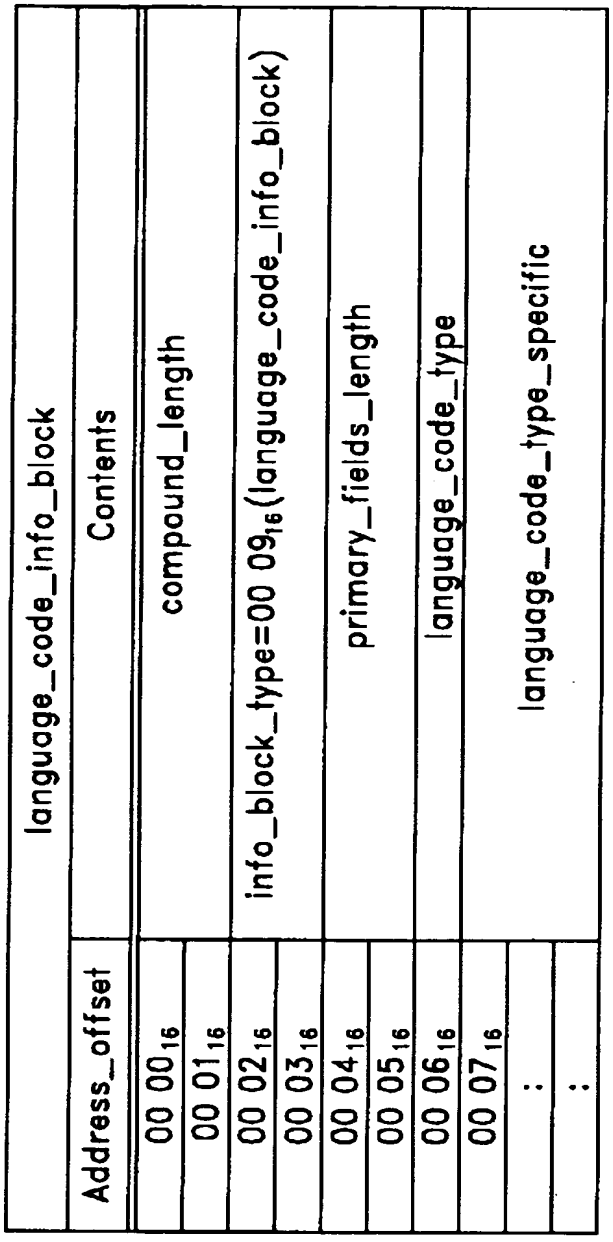
address_offset	contents
00_{16}	compound_length
01_{16}	
02_{16}	info_block_type
03_{16}	
04_{16}	primary_fields_length
05_{16}	
06_{16}	number_of_subunits
07_{16}	subunit_type_and_ID[0]
:	:

Subunit_Resource_info_block

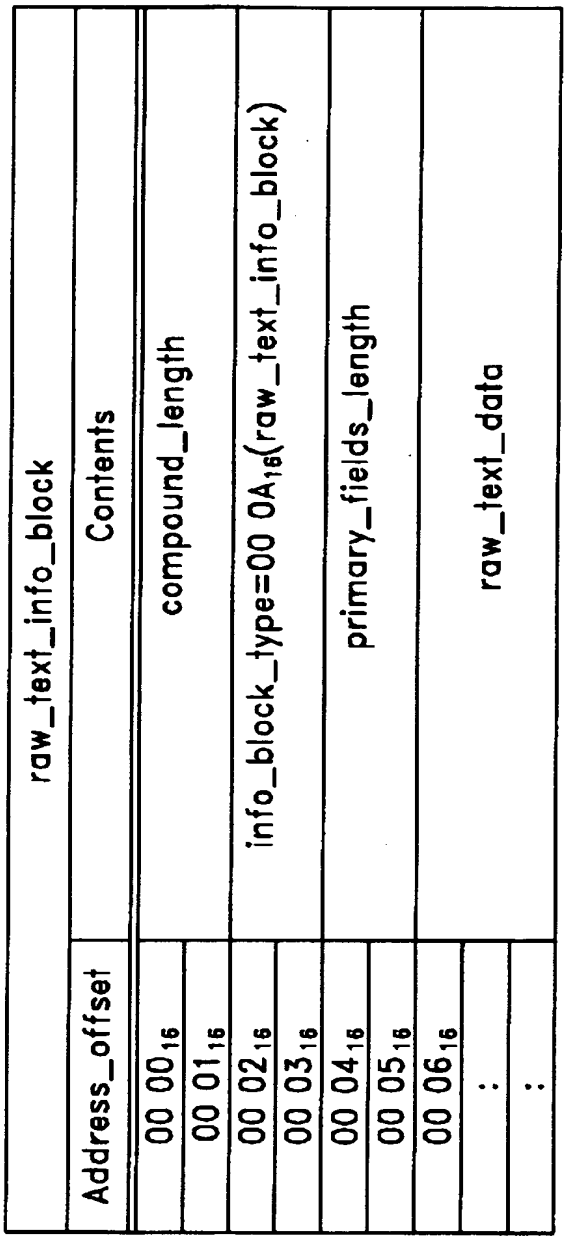
【図１４】


charactar_code_info_block
Address_offset
Contents
00 00$_{16}$
00 01$_{16}$
compound_length
00 02$_{16}$
00 03$_{16}$
info_block_type=00 08$_{16}$(character_code_info_block)
00 04$_{16}$
00 05$_{16}$
primary_fields_length
00 06$_{16}$
character_code_type
00 07$_{16}$
character_code_type_specific

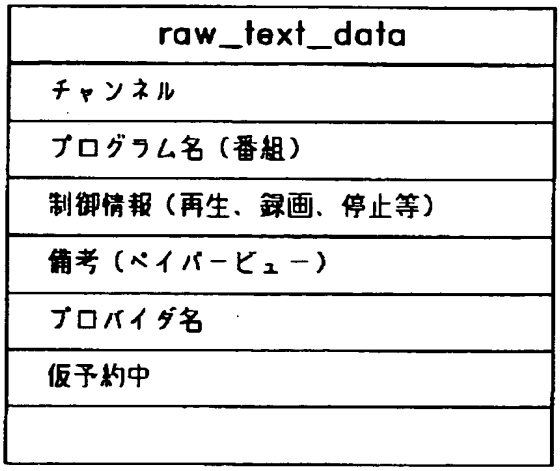
【図１５】


language_code_info_block
Address_offset
Contents
00 00₁₆
00 01₁₆
00 02₁₆
00 03₁₆
00 04₁₆
00 05₁₆
00 06₁₆
00 07₁₆
compound_length
info_block_type=00 09₁₆(language_code_info_block)
primary_fields_length
language_code_type
language_code_type_specific

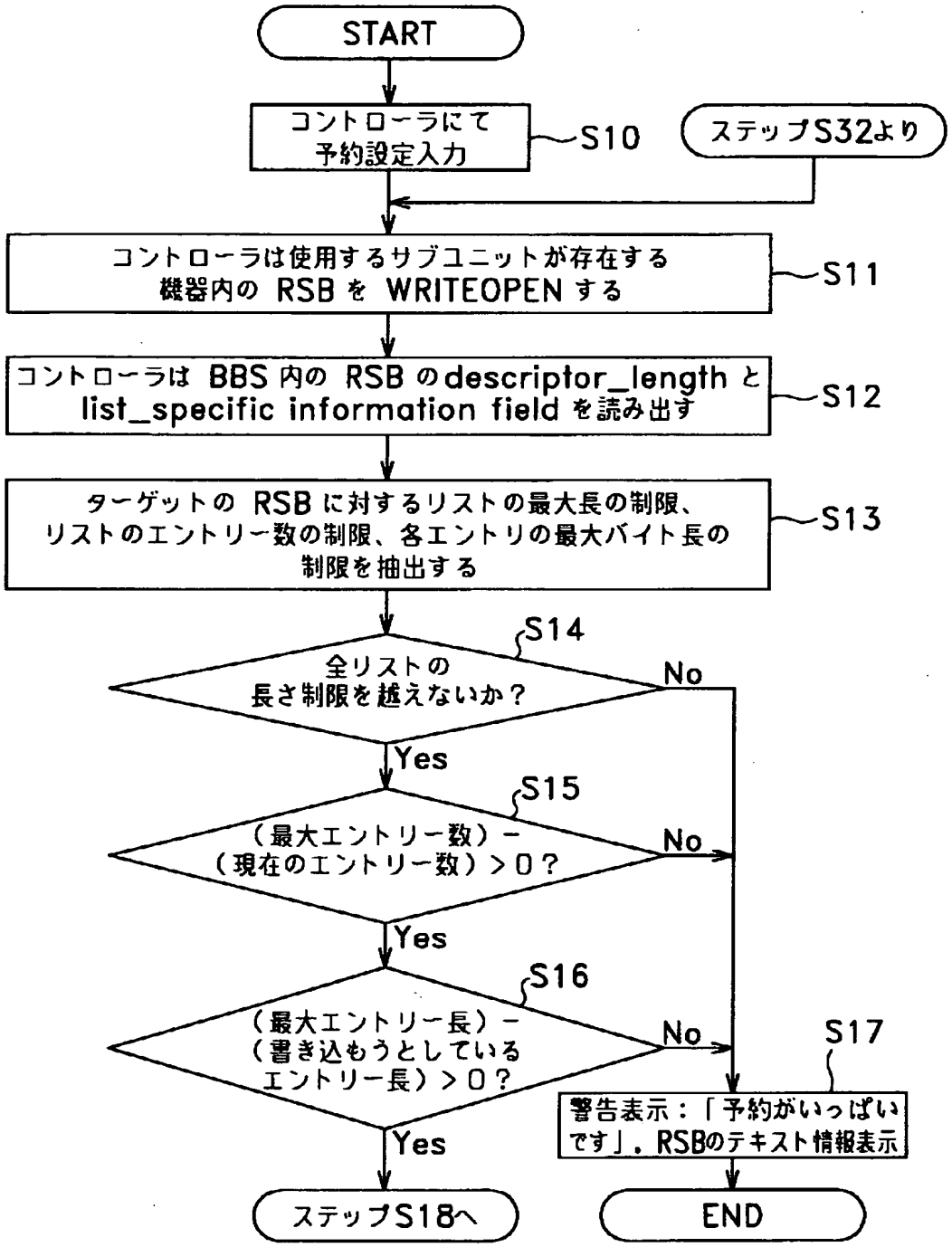
【図１６】


raw_text_info_block
Address_offset
Contents
00 00$_{16}$
00 01$_{16}$
compound_length
00 02$_{16}$
00 03$_{16}$
info_block_type=00 0A$_{16}$(raw_text_info_block)
00 04$_{16}$
00 05$_{16}$
primary_fields_length
00 06$_{16}$
raw_text_data

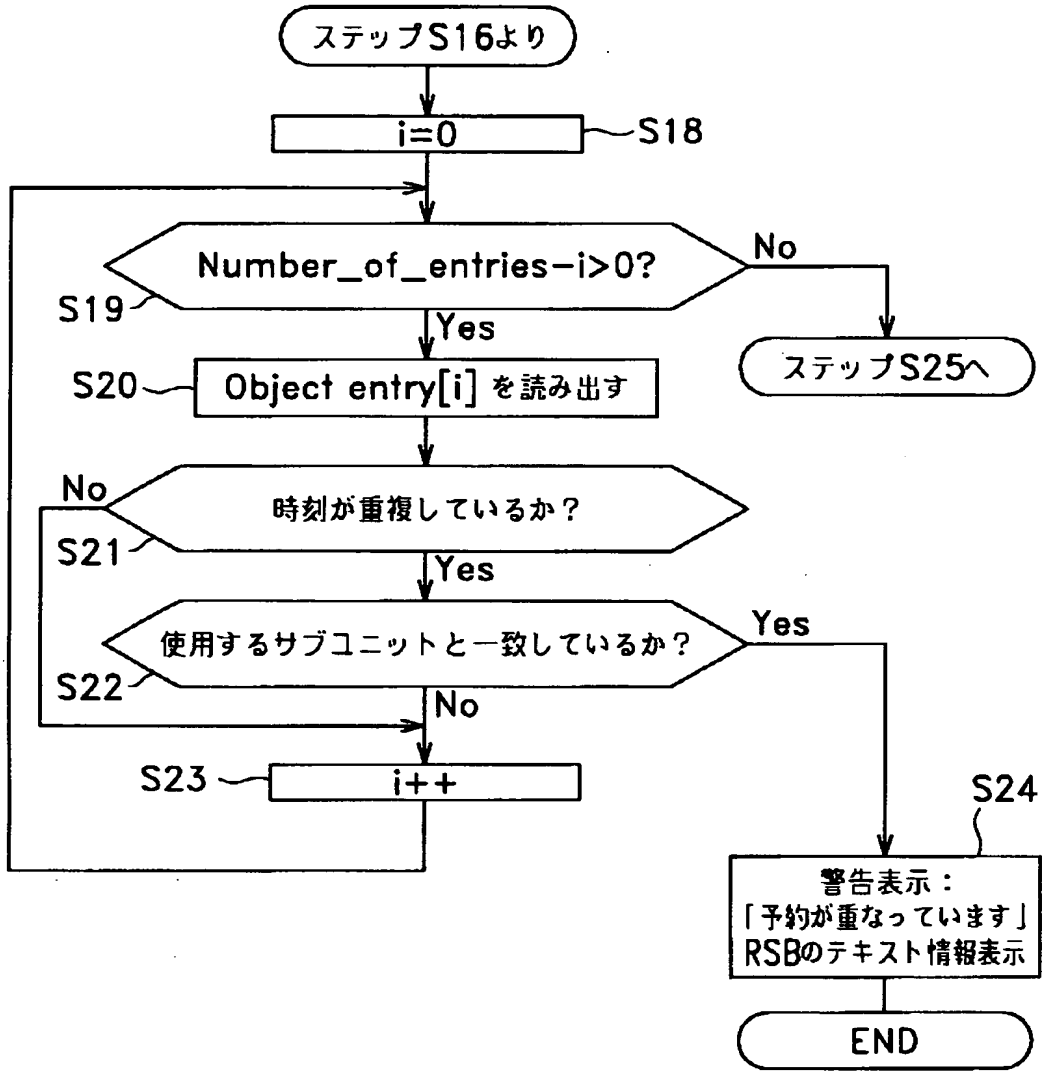
【図１７】


raw_text_data
チャンネル
プログラム名（番組）
制御情報（再生、録画、停止等）
備考（ペイパービュー）
プロバイダ名
仮予約中

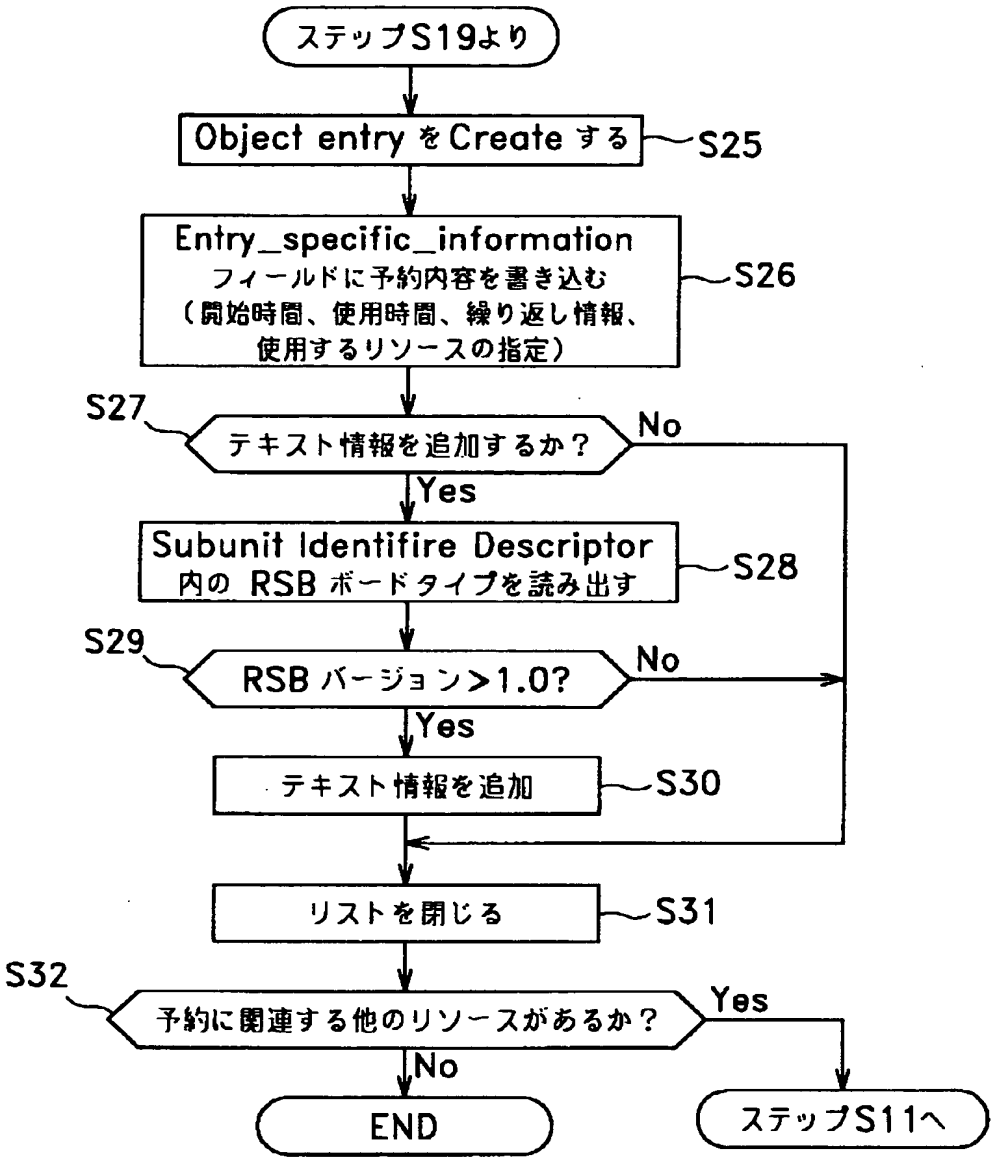
【図１８】


START
コントローラにて
予約設定入力
S10
ステップS32より
コントローラは使用するサブユニットが存在する
機器内の RSB を WRITEOPEN する
S11
コントローラは BBS 内の RSB のdescriptor_length と
list_specific information field を読み出す
S12
ターゲットの RSB に対するリストの最大長の制限、
リストのエントリー数の制限、各エントリの最大バイト長の
制限を抽出する
S13
S14
全リストの
長さ制限を越えないか？
No
Yes
S15
（最大エントリー数）－
（現在のエントリー数）＞０？
No
Yes
S16
（最大エントリー長）－
（書き込もうとしている
エントリー長）＞０？
No
S17
Yes
警告表示：「予約がいっぱいです」，RSBのテキスト情報表示
ステップS18へ
END

【図１９】


ステップS16より
i=0
S18
Number_of_entries−i>0?
No
S19
Yes
ステップS25へ
S20
Object entry[i] を読み出す
No
時刻が重複しているか？
S21
Yes
使用するサブユニットと一致しているか？
Yes
S22
No
S23
i++
S24
警告表示：
「予約が重なっています」
RSBのテキスト情報表示
END

【図２０】


ステップS19より
Object entry をCreate する
S25
Entry_specific_information
フィールドに予約内容を書き込む
（開始時間、使用時間、繰り返し情報、
使用するリソースの指定）
S26
S27
テキスト情報を追加するか？
No
Yes
Subunit Identifire Descriptor
内の RSB ボードタイプを読み出す
S28
S29
RSB バージョン>1.0?
No
Yes
テキスト情報を追加
S30
リストを閉じる
S31
S32
予約に関連する他のリソースがあるか？
Yes
No
END
ステップS11へ

【図２１】

WRITE OPEN コマンド

opcode	OPEN DESCRIPTOR
operand 0	descriptor_type=10_{16}
operand 1	List ID:00_{16}
operand 2	List ID:01_{16}
operand 3	subfunction WRITE OPEN 03_{16}
operand 4	reserved 00_{16}

【図２２】

READ コマンド

	msb							lsb
opcode	READ DESCRIPTOR (09_{16})							
operand 0	descriptor identifier							
operand 1	:							
:	:							
:	read_result_status							
:	reserved : 00_{16}							
:	data_length							
:	address							

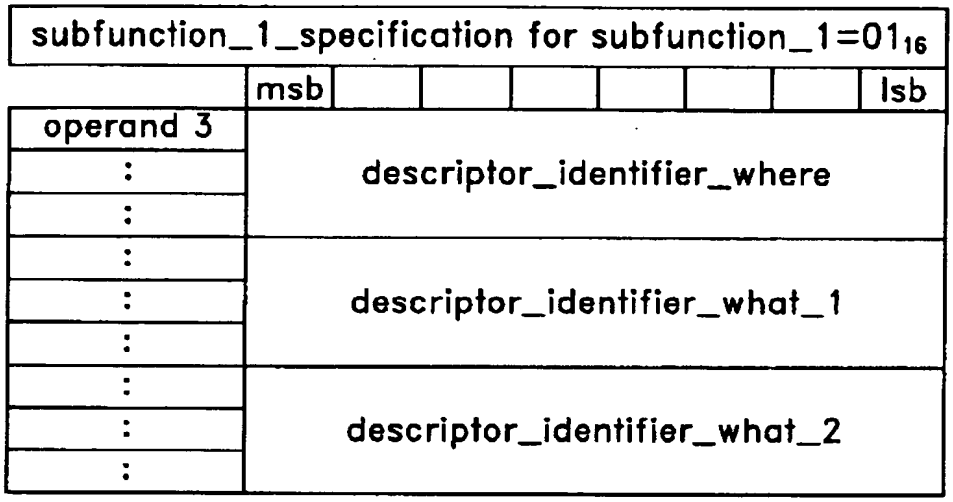
【図２３】

	msb							lsb
opcode	CREATE DESCRIPTOR (00_{16})							
operand 0	result							
operand 1	subfunction_1							
operand 2	reserved							
operand 3	subfunction_1_specification							
:								
:								

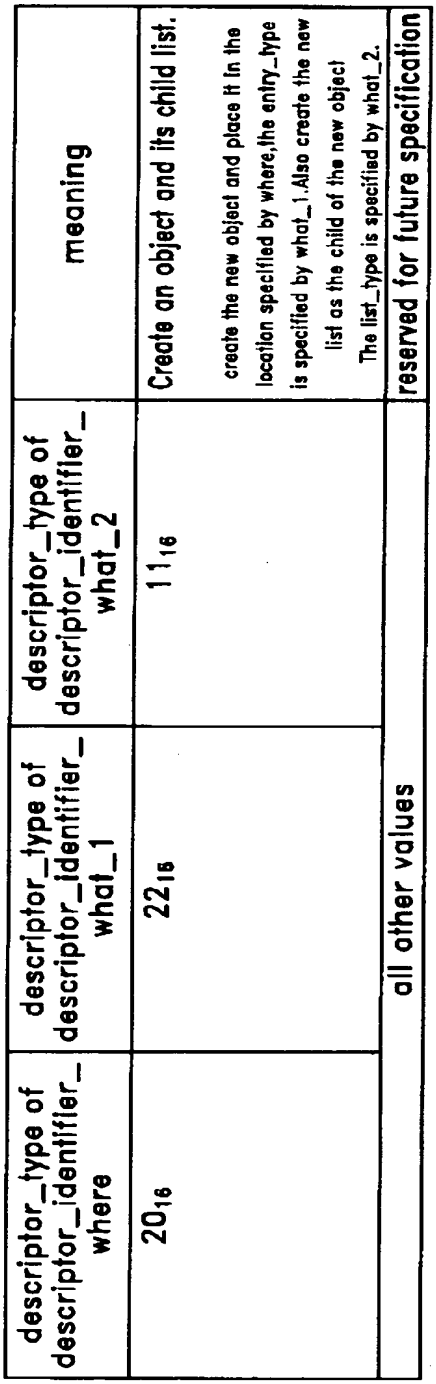
【図２４】

subfunction_1の値	meaning
00_{16}	create a new descriptior
01_{16}	create a new object and its child list
all other values	reserved for future specification

【図２５】


subfunction_1_specification for subfunction_1=01₁₆
msb
lsb
operand 3
descriptor_identifier_where
descriptor_identifier_what_1
descriptor_identifier_what_2

【図２６】



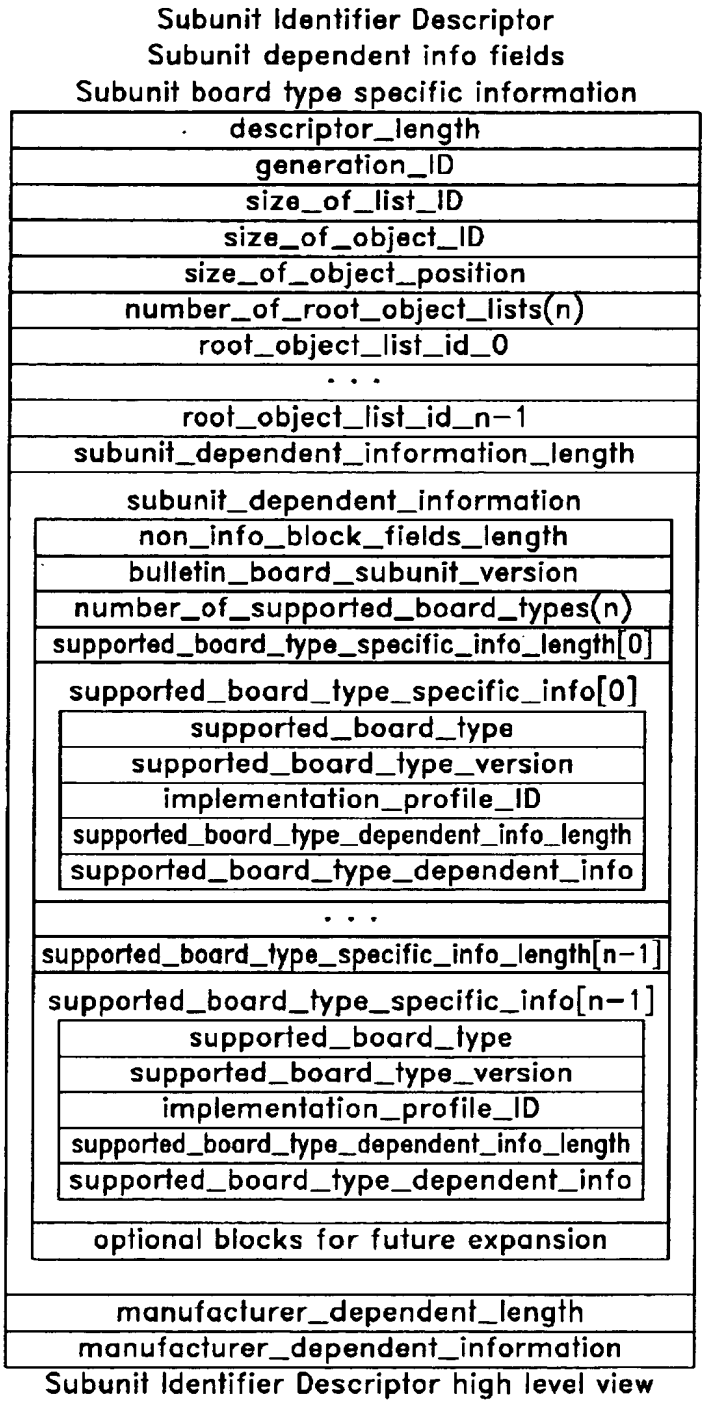
descriptor_type of descriptor_identifier_ where	descriptor_type of descriptor_identifier_ what_1	descriptor_type of descriptor_identifier_ what_2	meaning
20_{16}	22_{16}	11_{16}	Create an object and its child list. create the new object and place it in the location specified by where, the entry_type is specified by what_1. Also create the new list as the child of the new object The list_type is specified by what_2.
all other values			reserved for future specification

【図２７】

WRITE DISCCRIPTOR コマンド

opcode	WRITE DESCRIPTOR (OA_{16})
operand 0	descriptor identifier
:	subfunction:partial_replace(50_{16})
:	group_tag:immediate(00_{16})
:	replacement_data_length
:	address
:	original_data_length
:	replacement_data

【図２８】



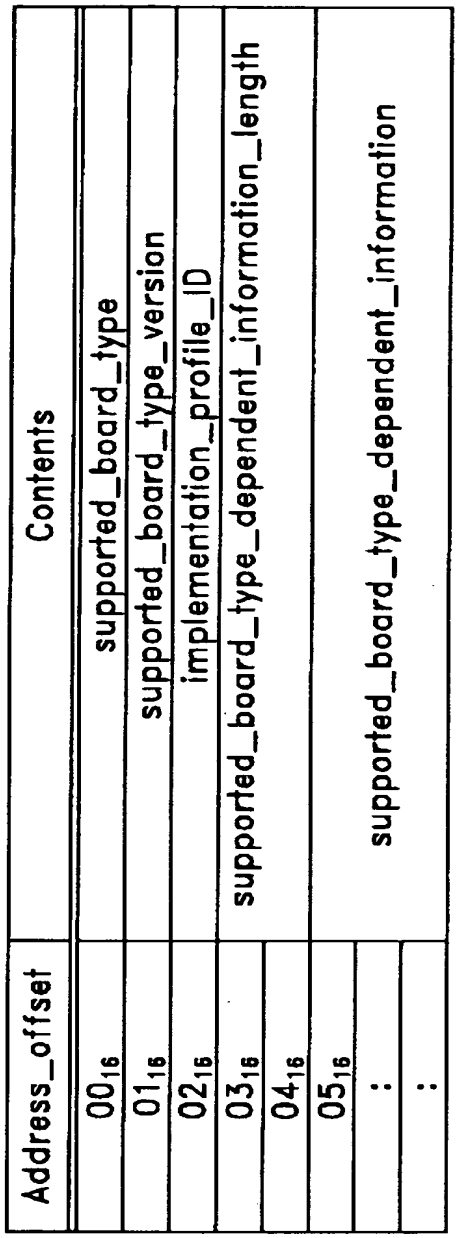
Subunit Identifier Descriptor high level view

【図２９】

root_object_list_ID Value Assignment

Value	List definition
1001_{16}	Resource Schedule List
$1002-10FF_{16}$	reserved

【図３０】



Address_offset	Contents
00_{16}	supported_board_type
01_{16}	supported_board_type_version
02_{16}	implementation_profile_ID
03_{16}	supported_board_type_dependent_information_length
04_{16}	
05_{16}	supported_board_type_dependent_information
:	
:	

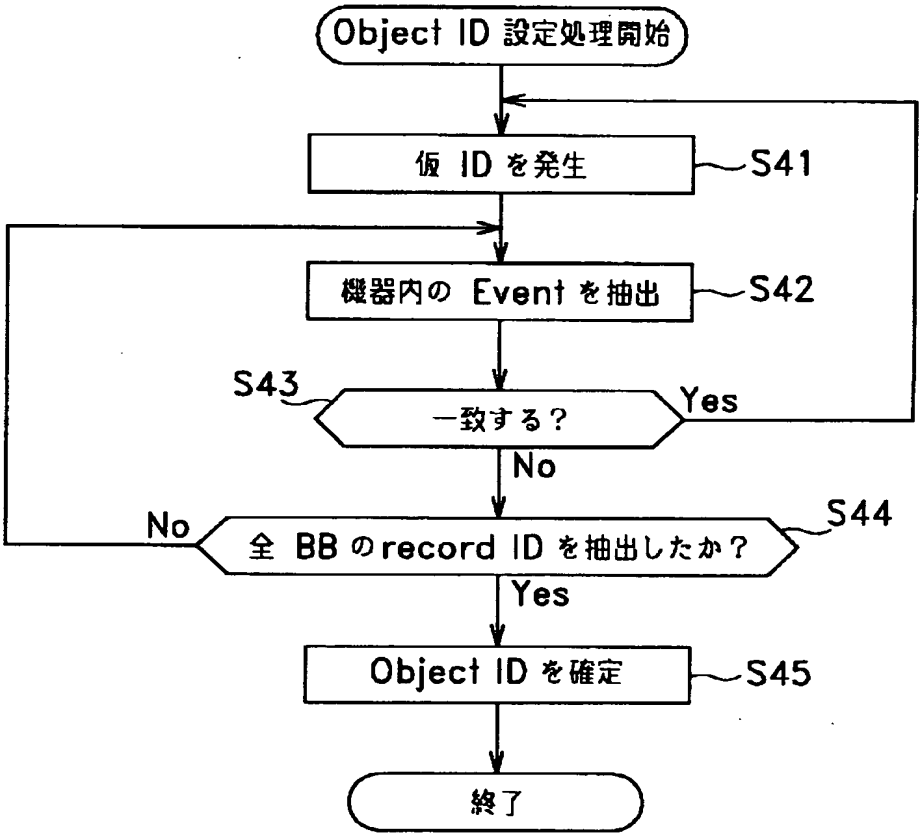
supported_board_type_specific_information fields

【図３１】

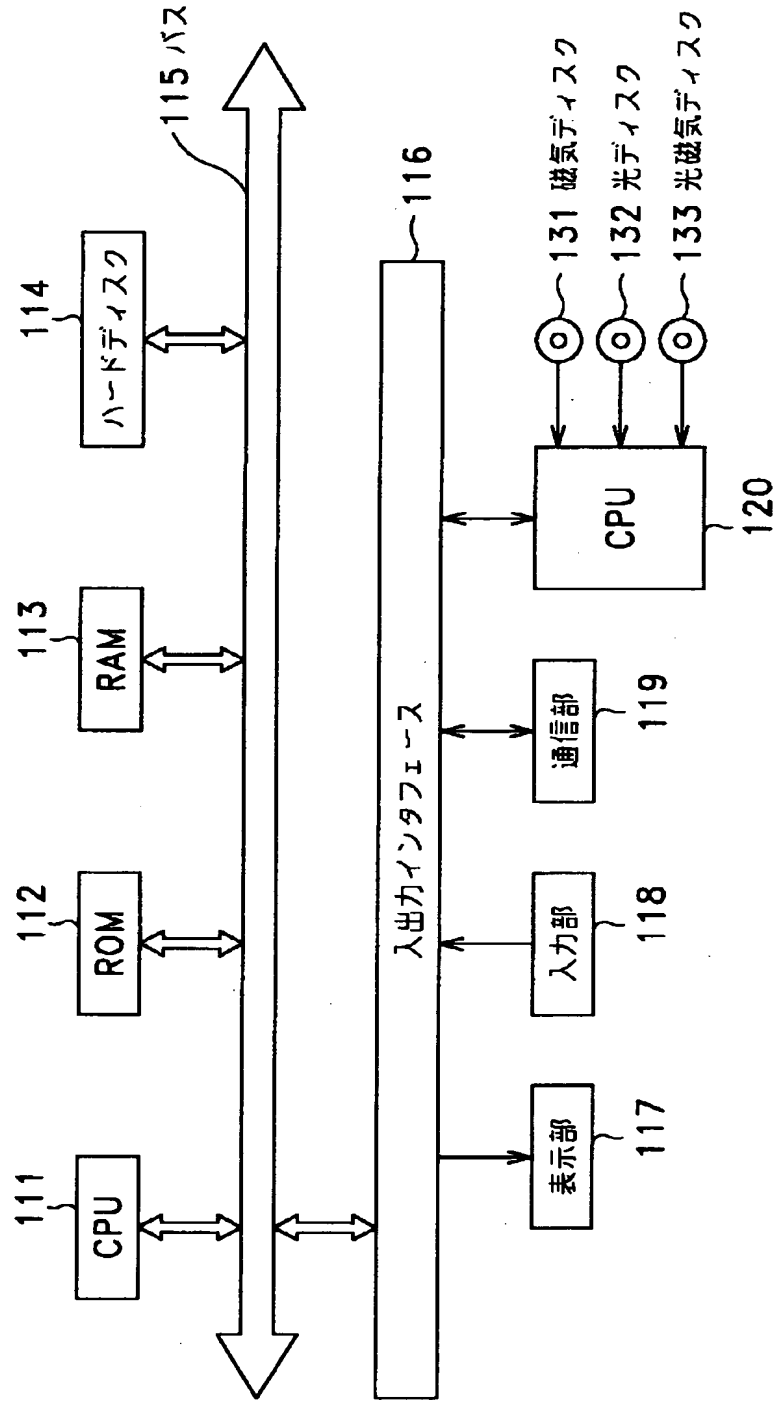
CLOSE コマンド

opcode	OPEN DESCRIPTOR
operand 0	descriptor type=10_{16}
operand 1	List ID:00_{16}
operand 2	List ID:01_{16}
operand 3	subfunction CLOSE 00_{16}
operand 4	reserved 00_{16}

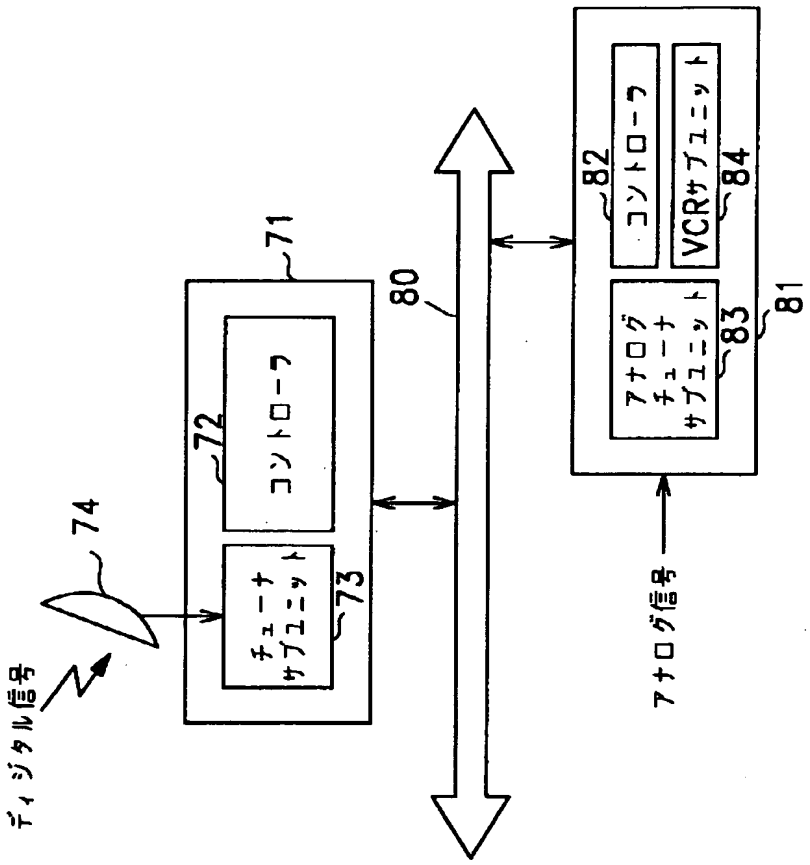
【図３２】



【図３３】


111
CPU
112
ROM
113
RAM
114
ハードディスク
115 バス
116
入出力インタフェース
117
表示部
118
入力部
119
通信部
120
CPU
131 磁気ディスク
132 光ディスク
133 光磁気ディスク
パーソナルコンピュータ 101

【図３４】


71
72
コントローラ
チューナ
サブユニット
73
74
ディジタル信号
80
82
コントローラ
VCRサブユニット
84
アナログ
チューナ
サブユニット
83
81
アナログ信号

【書類名】　　要約書

【要約】

【課題】　ユーザや各機器が予約情報等の詳細を入力でき、各機器が他の機器による予約情報等の詳細を入手でき、各機器が他の機器に対して自己が持つ予約情報等の詳細を知らせる手段を実現し、ダブルブッキングの発生を抑止し、分かり易く且つ使い易いネットワークシステムを構築可能とする。

【解決手段】　ＲＳＢのResource Schedule Entryに、予約の詳細を表すローテキストを含むraw_text_information_blockと、ローテキストのコーディングを表すcharacter_code_information_blockとlanguage_code_information_blockを記述する。

【選択図】　　図７

出　願　人　履　歴　情　報

識別番号　　〔０００００２１８５〕

１．変更年月日　　１９９０年　８月３０日
［変更理由］　新規登録
住　所　東京都品川区北品川６丁目７番３５号
氏　名　ソニー株式会社